7-7-2020

Molecular Distance Geometry Approach to solve Alpha Carbon Trace Problem

Lokesh Gupta
University of Windsor

Molecular Distance Geometry Approach to solve Alpha Carbon Trace Problem

by

Lokesh Gupta

A Thesis

Submitted to the Faculty of Graduate Studies

through the School of Computer Science

in Partial Fulfillment of the Requirements for

the Degree of Master of Science at the

University of Windsor

Windsor, Ontario, Canada

2020

Molecular Distance Geometry Approach to solve Alpha Carbon Trace Problem

by

Lokesh Gupta

APPROVED BY:

M. Hlynka
Department of Mathematics and Statistics

D. Wu
School of Computer Science

A. Mukhopadhyay, Advisor
School of Computer Science

May 19, 2020

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyones copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

To structural researchers, predicting protein structures currently remains a challenging task. During the past decades, different methodologies have been developed to address this issue. One such protein structure prediction problem is the Alpha Carbon (C_α) Trace Problem. The C_α trace problem is to determine the 3-D coordinates of the main chain atoms(C, N, and O) from just the C_α carbon coordinates. This master's thesis presents a novel approach for solving the C_α trace problem by using a molecular distance geometry approach.

The current approach uses the algorithms which are used to solve the Molecular Distance Geometry Problem to find the coordinates of the atoms in the peptide plane of a given protein. Once, the coordinates of the atoms(C_α, C, N, and O) in the single peptide plane are computed, the two C_α atoms are aligned with the first two C_α atoms in the C_α trace by finding the appropriate rotation and translation. The same rotation and translation are applied to all the other atoms in the peptide plane(C, N, and O). The process is then repeated for the entire trace, and the coordinates of all the atoms in the main chain of the protein are retrieved. In order to predict the side-chain atoms from the main Chain, SCWRL4.0 is used. The output generated by SCWRL4.0 is then subjected to LBFGS energy minimizer using a tool called MESHI.

The key advantage of using our approach is that it eliminates the building and searching for a huge protein fragment library. Experiments show that our approach is highly comparable to other approaches such as BBQ, PD2Main, and PULCHRA for solving the C_α trace problem.

DEDICATION

To my loving family, who has supported me in every step of my life.

ACKNOWLEDGMENTS

I express my sincere gratitude to Dr. Asish Mukopadhyay, without whose patient guidance and constant supervision, I would not have come so far.

I offer my sincere appreciation to the committee members, Dr. Myron Hlynka and Dr. Dan Wu for their useful critiques and advice.

My special thanks to my loving sister Dharna, who spent most of her time in active discussions and gave moral support that helped me to finish up my thesis.

My grateful thanks is also extended to my colleagues-cum-friends Md. Zamilur Rahman, Sudiksha, Aayushi, Saurav, Anjali, Jayanth, Parth and Dipesh for their invaluable help throughout my Master's degree. I would also like to thank my uncle Mr. Sanjiv Aggarwal and aunt Mrs. Meena Aggarwal for providing moral support during the course of my degree in Canada. Finally to my loving parents for their unmatched support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Distance Geometry [49] is the study of techniques on the set of points given only the distances between the pair of points. Nowadays, due to the several real-life applications, a large community of researchers are actively working in the field of distance geometry. One such application is in the field of telecommunication networks, where the distance between some sensors were known; the problem is to calculate the position of all sensors in space. Another intriguing application is in the field of biology, where the experimental techniques will measure the distance between a molecule's pair of atoms, and the question would be to determine a molecule's three-dimensional conformation.

Distance Geometry Problem can also be understood in the form of graph embedding problem given by Saxe[46]. The problem is formally defined as: Given an incomplete edge-weighted graph G and a parameter k, map the vertices of the graph G to the points in a Euclidean k-space in such a way for any two vertices connected by an edge, its edge weight is equal to the corresponding points in the k-dimensional space. Deciding if such an embedding exists is strongly NP-complete[46].

1.1 The Molecular Distance Geometry Problem

The Molecular Distance Geometry Problem (MDGP) tries to find the coordinates of the atoms of a molecule given only the subset of distances between the pair of atoms. Therefore, it can be termed as the three-dimensional version of Saxe's Problem. The MDGP finds its use case in NMR experimental techniques[5] which provides a set inter-atomic distance d_{ij} for a certain pairs of atoms (i, j) for a given molecule.

Formally, MDGP can be defined as: To determine a unique three-dimensional structure of a molecule when only the distances between all the pairs of atoms in that molecule are provided. However, if there are errors or certain distances are unavailable, the unique and correct structure of the molecule may not be calculated[8].

1.2 Preliminaries

This section provides some of the terminologies which are required before defining the actual problem statement of the thesis.

1.2.1 What is a Protein?

Proteins are long-chain molecules made up of amino acids. There are only 20 different kinds of amino acids that are present in proteins. Proteins are one of the living organism's most abundant organic molecules and have the most diverse functional spectrum of all the macromolecules. They perform a wide range of functions within humans, including catalyzing metabolic processes, replicating DNA, reacting to stimuli, providing structure to cells, and transporting molecules from one site to another and so on.

Amino Acids are the building blocks of a protein molecule. Each amino acid consists of a central carbon atom (C_α), hydrogen, a carboxyl group, and a variable R group. This R group is attached to C_α atom. The R group uniquely distinguishes which class of amino acids it belongs to, electrically charged hydrophilic side chains, polar but uncharged side chains or nonpolar hydrophobic side chains, and special cases. The structure of a typical amino acid is given below:

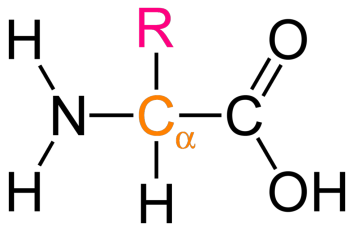


Figure 1.1: Structure of an amino acid [7]

The primary sequence of a protein is linked together using *dehydration synthesis*, which is defined as the loss of water molecule. This process combines the carboxylic acid of the upstream amino acid (Amino Acid - 1) with the amine functional group of the downstream amino acid (Amino Acid - 2). The amide linkage formed between two amino acids is called the peptide bond. Figure 1.2 shows the formation of a peptide bond.

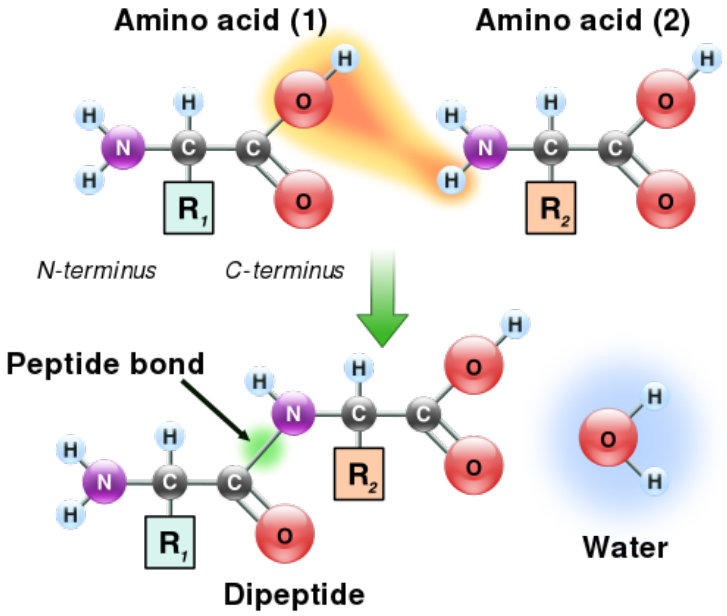


Figure 1.2: Synthesis of Protein [21]

1.2.2 The Peptide Plane

In proteins, an important structural feature is that all the five atoms (C_α, C, N, O and C_α) lie in the same rigid planar structure i.e., all these atoms are co-planar. This is because of the fact that there is a partial double bond character between the C and N atoms[40]. The structure of a peptide plane is shown in figure 1.3.

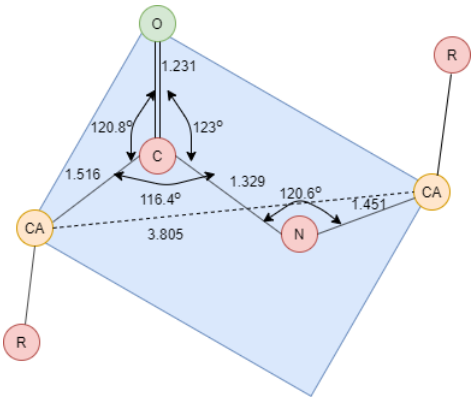


Figure 1.3: Structure of the Peptide Plane

From figure 1.3, we can also see the average value of the bond lengths and bond angles between the different atoms in the peptide plane. These values are published by Engh and Huber [15]. In proteins, two peptide planes are joined together at mutual C_α atoms, and each peptide plane can rotate about its bond to the C_α atom. The rotation around $C_\alpha - N$ bond is called phi(ϕ), and the rotation around $C_\alpha - C$ bond is called psi(ψ).

1.2.3 The Protein Data Bank(PDB)

The Protein Data Bank (PDB) [3] was first born at Brookhaven National Laboratories in the year 1971. This archive contained only seven structures of proteins in the beginning. The advent of new technologies such as Nuclear Magnetic Resonance(NMR) imaging[5] and X-ray crystallography[9] for structure determination in the year 1980s exponentially increased the number of available structure in the archive. A major role was played by the internet in making this bank highly accessible.

All the known and newly discovered protein structures are stored in this repository in the PDB format. The PDB format consists of data for each atom present in the structure, viz. its type and (x,y,z) coordinates, residue number, and the type of the residue. This information about each atom takes up a single line in the PDB file. For instance, an entry in the PDB file for the HYDROLASE which has PDB code 2LYM is as follows:

ATOM 99 CA ARG A -13.957 14.877 14.796 .....(1)

Similarly if there are two other atoms:

ATOM 110 CA HIS A -11.867 14.864 17.926 .....(2)

ATOM 120 CA GLY A -13.752 17.862 19.314 .....(3)

In short, a pdb file is a digitized record of the actual protein structure. The above (1) indicates that there is a carbon atom with the value of x, y, and z coordinates (-13.957,14.877,14.796). Moreover, the 'CA' shows that is the central C_α atom of a residue, namely residue 14 of type 'ARG' from chain A. The value 99 is a unique atom identifier within the file. Similarly, the other two are also shown in atom(2) and atom(3).

1.3 Problem Statement

Coarse-grained protein models (with some missing atomic details) are the result of many experimental or computational methods used for investigating a large number of protein structures and their dynamics. In this regard, there is an open problem called the C_α trace problem, which is the major focus of this thesis work.

To understand this problem, we must first take a quick look at how the structure of a protein is determined. One of the traditional techniques used for protein structure determination is X-ray crystallography[9]. This technique is time-consuming because we need to first crystallize the purified protein. Once the process of crystallization is complete, X-ray diffraction is used to determine the

electron density map of the crystal. The crystallographer is then able to determine the coordinates or positions of the constituent heavy atoms (other than hydrogen) of the protein crystal. Another popular technique is Nuclear Magnetic Resonance (NMR) spectroscopy[5]. This technique creates a graph consisting of peaks that correspond to the shift due to each nucleus in the molecule. The high-resolution structure is generated, but these structures are still subject to inaccuracies.

The C_α trace problem arises when we are provided with only estimated alpha carbon(C_α) atom coordinates or positions for a given protein, and we would like to determine the rest of the structure given that information alone. Figure 1.4 shows the snapshot of an incomplete C_α trace PDB file.

```
ATOM      1  CA  VAL A  24    -179.987-102.799 197.526  1.00525.53           C
ATOM      2  CA  ARG A  25    -180.167 -99.239 196.364  1.00540.17           C
ATOM      3  CA  TYR A  26    -177.257 -96.899 197.155  1.00443.53           C
ATOM      4  CA  ALA A  27    -178.217 -93.628 198.563  1.00543.41           C
ATOM      5  CA  THR A  28    -176.937 -90.009 197.673  1.00390.99           C
ATOM      6  CA  ALA A  29    -176.067 -88.349 196.012  1.00450.36           C
ATOM      7  CA  LEU A  30    -178.477 -86.888 198.291  1.00487.41           C
ATOM      8  CA  LYS A  31    -176.479 -87.607 201.312  1.00478.21           C
ATOM      9  CA  LEU A  32    -174.628 -84.718 199.862  1.00494.77           C
ATOM     10  CA  PHE A  33    -177.768 -83.228 198.770  1.00550.00           C
ATOM     11  CA  SER A  34    -179.839 -83.577 201.889  1.00445.88           C
ATOM     12  CA  GLY A  35    -176.750 -82.786 203.950  1.00422.64           C
ATOM     13  CA  GLU A  36    -177.119 -79.497 202.419  1.00512.42           C
ATOM     14  CA  VAL A  37    -180.919 -79.267 202.478  1.00481.98           C
ATOM     15  CA  PHE A  38    -180.920 -78.806 205.998  1.00473.53           C
ATOM     16  CA  THR A  39    -179.500 -75.266 205.317  1.00448.30           C
ATOM     17  CA  ALA A  40    -182.890 -73.926 206.035  1.00479.36           C
```

Figure 1.4: Snapshot of C_α trace PDB file

We aim to complete the above PDB file by adding other atoms, namely C, N, and O between the peptide plane formed between two successive C_α atoms. In order to generate the all-atom representation, we also need to add the atoms for side chains belonging to each residue in the C_α trace. Therefore, a snapshot of a complete PDB file is given below:

```
ATOM      1  N   LYS A   1       3.280  10.157  10.354  1.00 12.97           N
ATOM      2  CA  LYS A   1       2.412  10.448   9.190  1.00 10.68           C
ATOM      3  C   LYS A   1       2.396  11.956   9.013  1.00 18.97           C
ATOM      4  O   LYS A   1       2.437  12.742  10.001  1.00 17.43           O
ATOM      5  CB  LYS A   1       0.985   9.920   9.481  1.00 10.76           C
ATOM      6  CG  LYS A   1      -0.024  10.397   8.432  1.00 18.11           C
ATOM      7  CD  LYS A   1      -1.447  10.047   8.805  1.00 24.18           C
ATOM      8  CE  LYS A   1      -2.394  10.553   7.694  1.00 30.33           C
ATOM      9  NZ  LYS A   1      -3.796  10.480   8.184  1.00 42.79           N
ATOM     10  N   VAL A   2       2.395  12.368   7.774  1.00 17.25           N
ATOM     11  CA  VAL A   2       2.407  13.764   7.343  1.00 16.82           C
ATOM     12  C   VAL A   2       0.976  14.055   6.849  1.00 20.53           C
ATOM     13  O   VAL A   2       0.502  13.470   5.859  1.00 20.09           O
ATOM     14  CB  VAL A   2       3.512  14.056   6.335  1.00 14.85           C
ATOM     15  CG1 VAL A   2       3.549  15.544   5.933  1.00 24.21           C
ATOM     16  CG2 VAL A   2       4.927  13.755   6.848  1.00 16.87           C
```

Figure 1.5: Snapshot of complete PDB file

1.4 Motivation

The motivation of the problem lies in the number of applications that are associated with C_α trace problem. As we know that X-ray crystallography is used for determining the positions of the atoms of a protein molecule is expensive and time-consuming. Therefore, a number of files in the Protein Data Bank consist of protein structure where only the positions of C_α atoms are given. Thus, a method is required to calculate the precise location of all the atoms of a protein molecule.

There are some protein structure prediction techniques such as [16], which begins by generating C_α trace as the first step. To improve the quality of the structure predicted and complete it, we need a solution to C_α trace problem.

1.5 Thesis Organization

The list below presents the organization of the chapters, which makes up this thesis.
A brief description of the topics is also given that each chapter deals with.

- Chapter 2 gives a clear background knowledge of the protein structure prediction techniques used for solving the C_α trace problem. This includes four existing methods to solve the C_α trace problem. A detailed description of each method is given. The first three methods use the protein fragment library approach, whereas the latter method uses an analytical approach to determine the structure of a protein molecule.

- Chapter 3 provides a review of the existing distance geometry techniques and also includes the Crippen and Havel's algorithm in depth which forms the basis for solving molecular distance geometry problem.

- Chapter 4 describes the proposed approach and its inner workings and also shows the experimental results after applying our algorithm.

- Chapter 5 concludes the work done in this thesis and suggests some possible future research directions.
- Bibliography declares a detailed list of references from which have been used as a guide for this thesis.

Chapter 2

Literature review of approaches to the Alpha Carbon(C_α) trace problem

The problem of determining the structure of a protein molecule remains one of the most challenging tasks for computational chemists. We know that several experimental and computational methods are used for investigating protein structure generate coarse-grained protein models (with some missing atomic details). These coarse-grained modeling tools are highly efficient in terms of the time required to build such models[25]. But protein models should be complete so that it can be used for practical structure-based studies, including drug design and protein design. Thus, an integration between coarse-grained modeling tools and tools which can efficiently complete the protein models is required. Alpha carbon(C_α) trace is one such coarse-grained model of a protein, which consists of only the positions of C_α atoms.

In this chapter, we will review four techniques that are being used for determining the coordinates of all the other atoms of a protein molecule given just the coordinates C_α atoms. The three techniques namely PULCHRA[45], BBQ[18] and PD2 ca2Main[34] are dependent on building and using a large protein fragment

library for reconstructing the missing atoms. The fourth technique[43] gives an analytical approach for finding the coordinates of the missing atoms. All these four techniques are discussed in the following sections.

2.1 PULCHRA

PULCHRA (Protein Chain Reconstruction Algorithm)[45] is a tool which is used for reconstructing full-atom representation of a protein molecule from the C_α trace. This tool can be installed locally as a standalone program, which is written in the C programming language. It reads coordinates of the atoms of the protein molecule in PDB format and outputs full-atom PDB files. The method generally works in three steps: optimization of C_α positions, reconstruction of backbone and its optimization, and finally, reconstruction of side chains. Each step in the program is independent of one another so the user can choose which step to perform based on different applications.

In the first step, the positions of C_α atoms are optimized, which is done by removing irregular configurations. This is achieved by using the steepest-descent gradient minimization algorithm and a simple harmonic potential. The potential (V) consists of the following terms: pairwise $C_\alpha - C_\alpha$ distances, $C_\alpha - C_\alpha - C_\alpha$ virtual bond angles, C_α excluded volume, and the deviation from the initial positions [eq. (2.1)]

$$\begin{aligned} V = w_1 \textstyle\sum_{i=1}^{N-1} (d_{i,j+1} - d_0)^2 + w_2 \sum_{i=1}^{N-2} (\theta_{i,j+1,i+2} - \theta_0)^2 \\ + w_3 \textstyle\sum_{i=1}^{N-2} \sum_{j=i+2}^{N} (d_{i,j} - d_{ex})^2 + w_4 \sum_{i=1}^{N} (d_{i,i_0} - d_u)^2 \end{aligned} \tag{2.1}$$

where N is the number of C_α atoms; w_1, w_2, w_3 and w_4 are the weights which corresponds to the potential terms; $d_{i,i+1}$ is the distance between the ith and the i + *1*th C_α atoms and d_0 is the equilibrium $C_\alpha - C_\alpha$ distance equal to 3.8Å; $\theta_{i,i+1,i+2}$ is the virtual bond angle which consists of the ith, i + *1*th and the i + *2*th C_α atoms; $\theta_0 = 70^\circ$ is the equilibrium angle; d_{ex} is equal to 4Å; d_{i,i_0} is the distance between actual and initial C_α atom positions. The values of potential terms for

the weights w_1, w_2, w_3 and w_4 were calculated by hand and are equal to 1.0, 2.0, 10.0 and 0.5 respectively.

After running the steepest-descent minimization procedure, the resultant structure consists of $C_\alpha - C_\alpha$ distances and $C_\alpha - C_\alpha - C_\alpha$ virtual bond angles which are nearly close to native values. In a typical scenario, 100 minimization steps are required for the procedure to converge.

The second step is to perform the backbone reconstruction. This step is based on the method proposed by Milik *et al*[33] but is more refined than the original algorithm. The procedure requires four successive C_α atoms. Each of these four C_α atoms forms a fragment. These fragments are used to rebuild the peptide plane atoms between the second and third C_α atoms. Four distances between C_α atoms are calculated, namely the distances between the first and third (r_{13}), second and fourth (r_{24}), and first and fourth (r_{14}) as shown in figure 2.1.

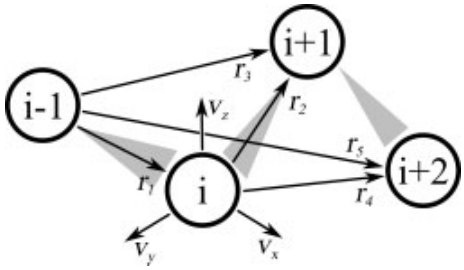


Figure 2.1: Frame of reference used for reconstruction of the backbone and side-chain atoms[45].

Now, r_{13} and r_{24} are divided into 10 bins where the distance lies between the range 4.5 to 7.5Åand r_{14} is divided into 75 bins where the distance lies between the range -11 to 11Å. Therefore, a lookup table is created using these bins, which is used to select proper fragments from the protein backbone fragment library to calculate the local coordinates of N, C, and O atoms that lies between the second and the third C_α atoms of the target fragment. The local system of coordinates is defined by three orthogonal axes v_x, v_y, v_z:

$$v_x = \frac{r_{13}}{|r_{13}|}$$

$$v_y = \frac{r_{23} \times r_{12}}{|r_{23} \times r_{12}|}$$

$$v_z = v_x \times v_y$$

Here r_{xy} is a vector which connects x and y C_α atoms. The reconstructed backbone often has a hydrogen bond pattern distorted. A simple optimization procedure is used to calculate the hydrogen bond energy of every peptide plate using the hydrogen bond definition found in the DSSP program[22]. The hydrogen bond (C-O...H-N) is rotated along with the $C_\alpha - C_\alpha$ virtual bonds, and the bond energy is calculated repeatedly, which is shown in figure 2.2.

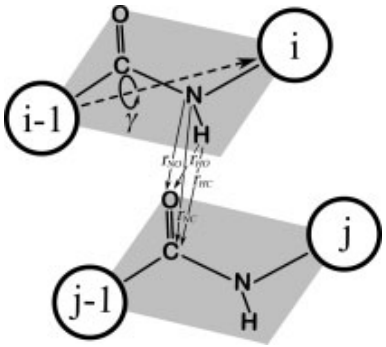


Figure 2.2: The procedure of hydrogen bond pattern optimization[45].

The energy function defined by the DSSP program[22] is described below:

$$E_{HB} = 332 q_1 q_2 \left(\frac{1}{r_{ON}} + \frac{1}{r_{CH}} + \frac{1}{r_{OH}} + \frac{1}{r_{CN}}\right) \tag{2.2}$$

where $q_1 = 0.42e$ and $q_2 = 0.20e$, with e being the electron charge unit, r_{XY} the distance between atoms X and Y. The energy is calculated and every time a better peptide plate orientation is found, the new orientation replaces the old one. This procedure repeats itself for every peptide plate.

The last step is to add the side-chains to predicted backbone atoms. Each bin that is present in the lookup table consists of a list of possible side-chain conformations. The conformation, which is nearest to the Center of Mass (CM), is used for reconstruction. If no CM is provided at the input, the side-chain conformation, which has the highest occurrence in the rotamer library, is used. This finally returns the full-atom model from the reduced C_α trace.

2.2 BBQ

The BBQ (Backbone Building from Quadrilaterals) method[18] adopts a similar approach for reconstruction of the protein backbone as proposed by Milik *et al*[33]. The dataset consisted of 1259 protein chains, which has mutual pairwise similarity not higher than 90%. This protein chain dataset was decomposed into fragments, which resulted in a library of 263,000 fragments. Figure 2.3 shows the protein backbone reconstruction flowchart.

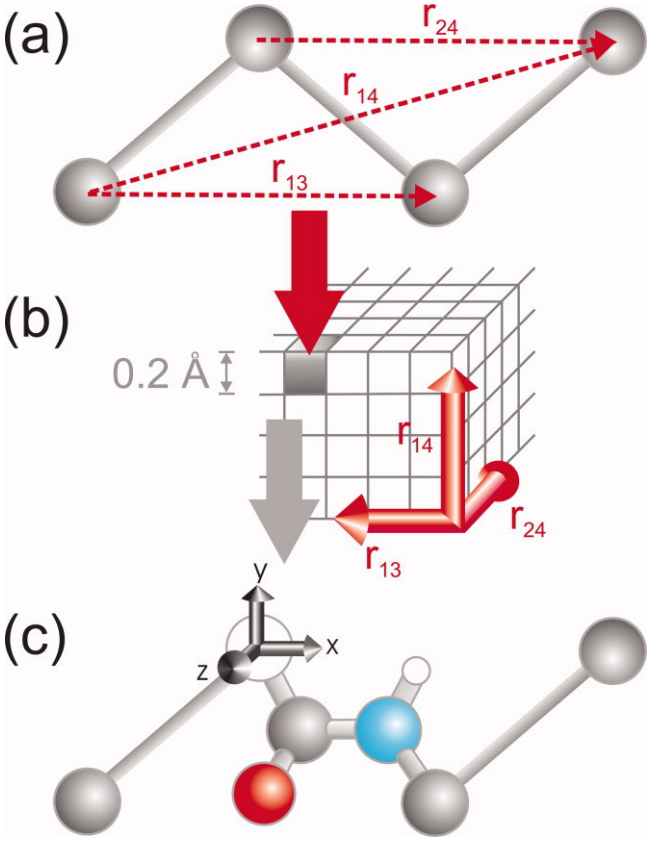


Figure 2.3: Protein backbone reconstruction flowchart[18]

The first step is to define two coordinate systems, namely the R-coordinate system and L-coordinate system. R-coordinates are computed as three distances, which are marked by the red dashed line, as shown in figure 2.3(a). R_{13} is defined as the distance between first and third C_α atoms, similarly R_{24} and R_{14} are defined. Here, each of the C_α atoms is treated as a point. Each continuous fragment consisting of four C_α atoms is called a quadrilateral. These R-coordinates are

used to describe a quadrilateral for each fragment. The second coordinates (L-coordinates) define a local Cartesian coordinate system, which is centered on a given C_α atom. Therefore, for each quadrilateral, L-coordinates are defined as a system of simple linear combinations:

$$\vec{v_x} = (\vec{v_{12}} + \vec{v_{23}})/|\vec{v_{12}} + \vec{v_{23}}|$$
$$\vec{v_y} = (\vec{v_{12}} - \vec{v_{23}})/|\vec{v_{12}} - \vec{v_{23}}|$$
$$\vec{v_z} = \vec{v_x} \times \vec{v_y}$$

where $\vec{v_x}$,$\vec{v_y}$ and $\vec{v_z}$ are the L-coordinates vectors and $\vec{v_{ij}}$ denotes a vector pointing from ith to jth C_α atom. The L-coordinates define the local positions of the backbone atoms.

For each of these quadrilaterals, the L-coordinates are computed for the atoms which form the central peptide i.e., the atoms lying between the second and third C_α atoms. R-coordinates are also computed and were divided 0.2 and rounded to the nearest integer. This resulted in discretized space defined by R-coordinates. Now, all the quadrilaterals which are defined by R-coordinates are stored in the lookup table. Also, the average grid positions of N, C, and O backbone atoms defined by L-coordinates are also computed for each of the grid elements, as shown in figure 2.3(b).

In the final reconstruction step, the R-coordinates of the target fragments are calculated. These R-coordinates are used to find a quadrilateral which is already stored in the lookup table. The retrieved quadrilateral also consists of the proper set of local coordinates for N, C, and O atoms. There are some rare cases in which a quadrilateral defined by a particular combination of R-coordinates cannot be found in the entire lookup table. In these cases, the algorithm inspects the neighborhood. If all the neighborhood is empty, the algorithm checks all quadrilaterals in the database and the entry which minimizes the distance r^{QD} (eq.2.3) between the R-coordinates of the query and an element in the lookup table(D):

$$r^{QD} = \sqrt{(R^Q_{13} - R^D_{13})^2 + (R^Q_{24} - R^D_{24})^2 + (R^Q_{14} - R^D_{14})^2} \tag{2.3}$$

2.3 PD2ca2main

PD2ca2main[34] is based on constructing Structural Alphabet (SA) in order to solve the C_α trace problem. A Structural Alphabet (SA) is a short library of motifs that together are able to describe most of the protein conformational space[37, 44]. A motif is defined as the smaller, and similar three-dimensional structures present the whole protein molecule, which performs a similar function. Therefore, Structural Alphabet represents a three-dimensional protein structure as a series of one-dimensional "letters"[38, 39]. Once the Structural Alphabet is constructed, it is used to find the missing backbone atoms and build a full backbone model.

The novelty in this work lies in the use of Gaussian mixture models (GMMs) for constructing this Structural Alphabet. A Gaussian mixture model is a probabilistic model that represents normally distributed subpopulations within an overall population. Here, the mixture model does not require any knowledge regarding a data point belonging to which subpopulation. Therefore, GMM is an unsupervised form of clustering technique. For building a structural alphabet using GMM, a large number of high-resolution PDB structures were used, which were decomposed into fragments. Several fragment sets were built for comparison, ranging in length from 4 to 7 consecutive C_α atoms, but the comparisons suggested that 6-mer fragments are the most suitable in this scenario. This resulted in a library of 480,000 fragments from the training PDB structures. An example of a typical fragment is shown in figure 2.4.

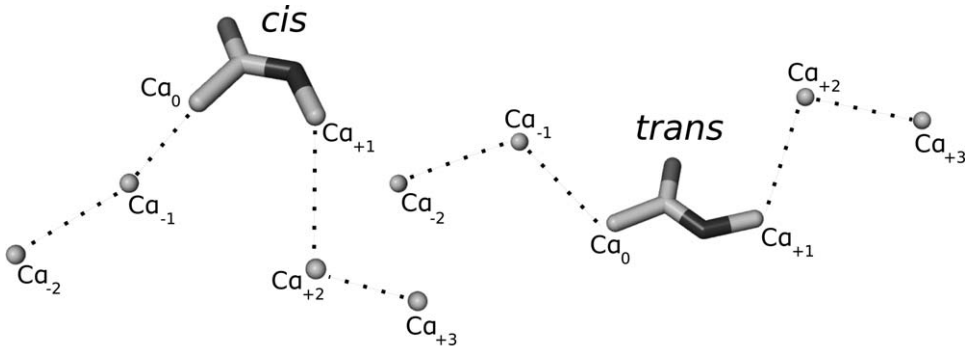


Figure 2.4: Example cis and trans components of a 6-mer fragment[34].

A GMM is used on the library of fragments generated where each fragment is of length 6 i.e., six consecutive C_α atoms. This is achieved by converting the fragment library to a dataset of 12-dimensional data points. Using the figure 2.4 as a reference, a single point in the dataset is represented as:

$$x = (C^x_{\alpha-2}, C^y_{\alpha-2}, C^z_{\alpha-2}, \ldots, C^x_{\alpha+3}, C^y_{\alpha+3}, C^z_{\alpha+3})$$

Since the $C_{\alpha 0}$ and $C_{\alpha+1}$ are adjacent to the fixed idealized peptide bond, this results in a minimal variation in their positions. Therefore, this information is not included in the data used by GMM fitting.

The GMM fitting to this dataset started with hierarchical clustering of data points which gives starting values for the expectation-maximization (EM) algorithm. Since a large number of points are present in the beginning, only 2000 random sample points were chosen to perform the initial clustering. These clusters then initialize the iterative rounds of EM, which used the complete dataset until the convergence is reached.

The Bayesian information criterion (BIC) was used to evaluate the optimum number of model components or clusters. While fitting a model using GMM, it is possible to add parameters in order to increase the likelihood of a model, but this can overfit the model. To avoid this, BIC introduces a penalty term for the number of parameters in the model. Since these penalty terms are stronger than the Akaike information criterion (AIC), it is particularly more suitable in this instance. There is always a trade-off between the number of components resulting from the model and the complexity of the model. It is expected that a large number of components would result in better performance, but this will increase the complexity of the model. Thus, we need a model that is reasonably constrained in order to retrieve an alphabet small enough to maintain a fast reconstruction. In this scenario, BIC leads to the selection of 528 component mixture model.

To find the coordinates of the actual structure given a target C_α trace, the structure is first divided into 6-mer fragments. Now we are given a set of target 6-mer fragments and Structural Alphabet (SA) created before using the GMM. The

member of the Structural Alphabet (SA) which minimizes the weighted C_α RMSD superposition value using a fast quaternion-based method[30, 48] is determined. The Weight Factor (W) was successively decreased by a factor of ten for C_α atoms which are present at the outer positions such that

$$W(C_{\alpha-3}) = \frac{1}{100},\ W(C_{\alpha-2}) = \frac{1}{10},\ W(C_{\alpha-1}) = 1$$

$$W(C_{\alpha 0}) = 1,\ W(C_{\alpha+1}) = \frac{1}{10},\ W(C_{\alpha+2}) = \frac{1}{100}$$

The significant advantage of using this type of weighting scheme is to reduce the impact of averaging errors, which can distort the geometry of the outermost C_α atoms in some members of the alphabet. The factor of ten, which is used for the above weighting scheme, is not optimized thoroughly but gave the results which are adequate. The rotation matrix, which is determined by fitting the C_α atoms of the optimal "letter" or component, is then utilized to find the final placement of the peptide bond atoms on the target structure.

In addition to the EM algorithm used by the Gaussian Mixture Modelling technique to find the appropriate number of components, this software also provides an optional gradient energy minimization procedure, which can further improve the result of the computed structure. This can lead to the longer running time of the overall procedure. During energy minimization, the positions of C_α atoms are kept fixed while the other backbone atoms are free to move. This minimization procedure is performed using a previously defined simple backbone potential energy function[32].

This complete algorithm was implemented as part of a protein structure modeling software package called PD2 and was entirely written in the C++ programming language.

2.4 Analytical approach for protein backbone reconstruction

This section presents an analytical method for generating the entire backbone of the protein structure using only the coordinates of the C_α atoms [43]. This method has a purely analytical foundation since it uses only the trigonometric relations that exist between different bond lengths and bond angles in a protein molecule. There are two major steps involved in the reconstruction of the protein backbone (i) Determination of β-carbon positions (ii) Backbone reconstruction. These steps are briefly explained in the following subsections.

2.4.1 Determination of β-carbon positions

The first step in the reconstruction procedure is to determine the position of the β-carbon atoms. A C_β atom is defined as the first atom of the side chain in an amino acid[1]. The bond between α and β carbon atom is fixed between each residue. There is a need to determine the appropriate reference system which can uniquely find the position of the C_β atom. Such a reference system is defined using figure 2.5.

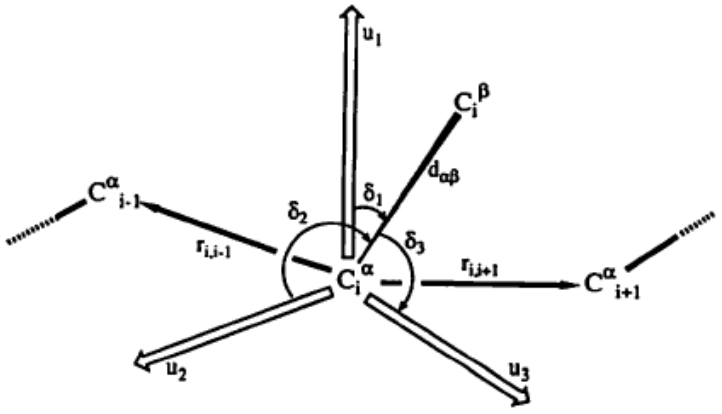


Figure 2.5: Schematic Representation of reference coordinate system to fix the position of C_β atom[43].

The mathematical equations for the reference system can be stated as:

$$\rho_{i,i+1} = \frac{r_{i,i+1}}{|r_{i,i+1}|} \qquad \rho_{i,i-1} = \frac{r_{i,i-1}}{|r_{i,i-1}|}$$

$$u_1 = \frac{\rho_{i,i+1} \times \rho_{i,i-1}}{|\rho_{i,i+1} \times \rho_{i,i-1}|}$$
$$u_2 = -\frac{\rho_{i,i+1} + \rho_{i,i-1}}{|\rho_{i,i+1} + \rho_{i,i-1}|}$$
$$u_3 = u_1 \times u_2$$

Here $r_{i,i+1}$ and $r_{i,i-1}$ are defined as the two vectors joining C_i^α with C_{i+1}^α and C_i^α with C_{i-1}^α respectively (refer Fig 2.5). Moreover, u_1, u_2 and u_3 are the reference axis.

Now the position of C_i^β with respect to C_i^α is determined by knowing the distance between the chemical bond C_α and C_β, $d_{\alpha\beta}$, and three direction cosines δ_1, δ_2 and δ_3 between this bond and the reference system defined above. The average values of distance between C_α and C_β, $d_{\alpha\beta}$, and three direction cosines δ_1, δ_2 and δ_3 are summed up in a table. These values are defined for each of 20 residues that are possible.

2.4.2 Backbone reconstruction

Once the coordinate of C_i^β for a given residue, is determined using the above step, the coordinates of Carbon (C) and Nitrogen (N), connected to the C_i^α atom, are determined. Refer to figure 2.6 for a full representation of a protein backbone that defines different bond lengths and bond angles between different atoms.

To compute the coordinates of C and N, the distance between C_i^α and C ($d_{\alpha C}$) and the distance between C_i^α and N ($d_{\alpha N}$) are assumed to be known. Along with this the angles $\tau_{N\alpha\beta}$, $\tau_{\beta\alpha C}$, $\tau_{N\alpha C}$ are also assumed to be known. These distances and angles are depicted in figure 2.6. All these values were found, for each residue, by analyzing the Protein Data Bank (PDB).

Several geometrical constraints are applied in the form of mathematical equations. The various geometrical constraints under which these equations lies are:

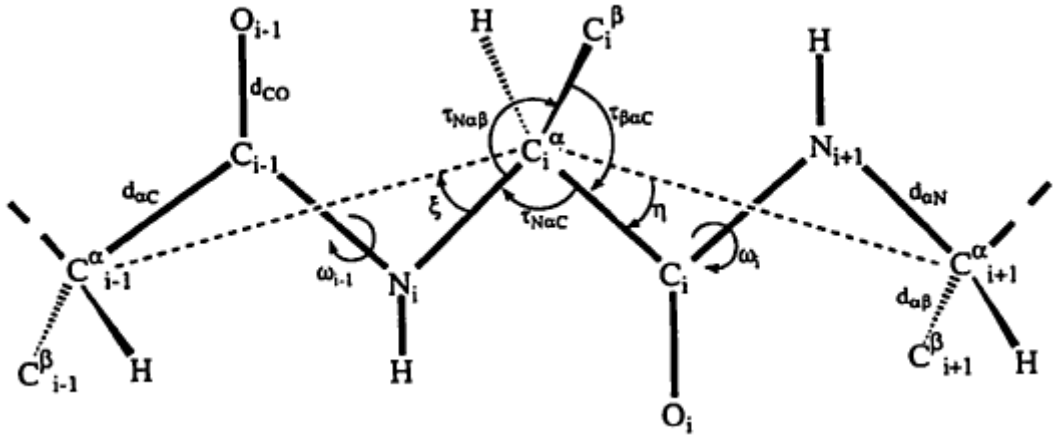


Figure 2.6: Full atom representation of protein backbone[43].

(i) Bond Angle, (ii) Bond Length, (iii) C_α chirality and (iv) Angles η and ϵ. All these equations must be satisfied simultaneously to find the coordinates of C_i and N_i atoms connected to C_α^i. For a detailed description of these mathematical equations, please refer to [43]. The solution to these equations leads to the coordinates C_i and N_i atoms.

By knowing the coordinates of C_i, one can easily determine the coordinates of O_i by using the distance between C_i and O_i (d_{CO}) and the angle $\tau_{O\alpha\alpha}$. The angle is defined as the angle between atom O, atom C_α^i and atom C_α^{i+1}. This process is repeated for all the C_α atoms in the protein molecule and the coordinates of all the atoms of the protein backbone are calculated.

Chapter 3

Distance Geometry approach

Distance Geometry problem is to find the coordinates of a set of points when only distances between some pairs of points are provided. A lot of applications of Distance Geometry problem lies in the field of biology where experimental techniques such as Nuclear Magnetic Resonance(NMR)[5] can measure the distances between pair of atoms of a given molecule. The problem to identify the three-dimensional structure of the molecule is called the Molecular Conformation problem. The major focus is on proteins because the three-dimensional structure of the protein provides clues about the functioning of that particular protein. Since the problem of Distance Geometry is used in the domain of a molecule, it is often termed as the Molecular Distance Geometry problem. In this thesis, the distance geometric approach is considered for solving the C_α trace problem, which does not involve the use of building huge protein fragment libraries.

3.1 Review of the Distance geometry techniques

This section will provide a thorough discussion of the algorithms and techniques[50], which are used for solving the Distance Geometry Problem. Also, an overview of some software packages which are built using these techniques is mentioned. The

first two techniques are concerned with finding the solution to the problem knowing all the exact distances between the points. The third technique reduces the problem into smaller subproblems, which can be solved to get a solution. The least-square minimization algorithm, which solves the distance geometry problem as a type of optimization, is discussed towards the end of the section.

3.1.1 Cayley-Menger Determinant

Let us assume that all the exact distances between all the pairs of points are provided then the mandatory conditions for the distance matrix

$$D(p_0, ..., p_n) = \begin{pmatrix} 0 & d_{01} & ... & d_{0n} \\ d_{10} & 0 & ... & d_{1n} \\ ... & ... & ... & ... \\ d_{n0} & d_{n1} & ... & 0 \end{pmatrix}$$

of *n+1* points $p_0, p_1, ..., p_n$ which can be embedded in euclidean space E^n is given by Cayley-Menger [47] such that the CM determinant$(p_1, ...p_n) \geq 0$. The given distance matrix for the points p_0, p_1, p_2, p_3 can be represented in the form of Cayley-Menger determinant[47]:

$$\begin{vmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & d_{01}^2 & d_{02}^2 & d_{03}^2 \\ 1 & d_{10}^2 & 0 & d_{12}^2 & d_{13}^2 \\ 1 & d_{20}^2 & d_{21}^2 & 0 & d_{23}^2 \\ 1 & d_{30}^2 & d_{31}^2 & d_{32}^2 & 0 \end{vmatrix}$$

The rank of the Cayley-menger matrix gives the minimum embedding dimension. For details please refer to [51].

One important use of Cayley-Menger matrix is to find the missing distances present in the distance matrix. For an arbitrary graph with n vertices, the pre-distance matrix D = $[D_{ij}]$ is a symmetric matrix such that $D_{ij} = d_{ij}^2$, where d_{ij} is the distance between the vertices (points) i and j of the arbitrary graph. The

Cayley-Menger matrix, C = $[C_{ij}\ ]$ is a symmetric $(n+1) \times (n+1)$ matrix such $C_{0i} = C_{i0} = 1$ for $0 < \ i \leq n$, $C[0,\ 0] = 0$ and $C_{ij} = D_{ij}$ for $1 < i, j \leq n$ [14].

3.1.2 Decomposition of Distance matrix

Suppose all the exact distances between all pair the points are known, then we can represent this in a form of a matrix, $D = [d_{ij}]$, where d_{ij} corresponds to the distance between points i and j. Let's assume we have set of points $x_0, x_1, x_2, ..., x_n$. We also assume that coordinates of x_0 to be (0,0,0) to be the origin.

Our problem is to calculate coordinates of these points. The distance constraints are given as:

$$|x_i - x_j| = d_{ij}, \quad i, j = 1, ...n$$

or equivalently

$$|x_i|^2 = d_{i0}^2$$

where d_{i0}^2 is distance between point i and origin

$$|x_i - x_j|^2 = d_{ij}^2$$

by expansion

$$d_{i0}^2 - d_{ij}^2 + d_{j0}^2 = 2{x_i}^T x_j, \quad i, j = 1, 2, ..., n$$

Let

$$D_{ij} = (d_{i0}^2 - d_{ij}^2 + d_{j0}^2)/2,$$

then a matrix D can be defined as

$$D = [D_{ij}]$$

Let X be an $n \times 3$ coordinate matrix

$$X = [x_1^T; .....; x_n^T;]$$

we have

$$D = XX^T$$

For a solution to exist the rank of matrix D must be 3. Therefore, we can perform a singular value decomposition for D to get

$$D = U\sigma U^T$$

Where U is an $n \times 3$ orthogonal matrix and σ is the eigen value diagonal matrix with diagonal elements represented by $\sigma_1, \sigma_2, \sigma_3$. These diagonal elements are three non-zero singular values of D. A solution for

$$D = XX^T$$

exists and is given as

$$X = U\sigma^{(1/2)}$$

The time complexity of singular value decomposition is $O(n^3)$ time. Therefore, a polynomial time solution to the distance geometry problem can be obtained given all the exact distances. More details can be found in [8].

3.1.3 Graph Reduction

Let's consider the distance geometry problem in terms of the graph embedding problem where points are considered as nodes and distances as edges. The weights on the edges are the distance values. Now the solution to the problem is to embed the graph in a Euclidean space. This is called the graph embedding problem. The graph is more often not a complete graph, meaning that some edges are missing. In such a case, there will not be a unique embedding. In other words, there is more than one way to position the points in euclidean space such that distance constraints are still satisfied. This type of graph is called a flexible graph.

The property of rigidity of a distance graph is important for studying the distance geometry problem. A mandatory condition that a graph has a unique embedding is that it must be rigid. A graph that has partial reflections is also not said to have unique embedding. This condition can only be ensured if, for three-dimensional embedding, a graph is four-connected. These conditions are used to find graphs or subgraphs that have unique embeddings. In order to solve the embedding problem for a given distance graph, decompose the distance graph into such sub-graphs. Once the solution for subgraphs is found, they can be combined to form a solution for the whole graph. For more details, refer to [20].

3.1.3.1 ABBIE

The ABBIE software package Hendrickson developed by [20] can be used to obtain the three-dimensional embedding of a molecular structure by inputting just the pairwise distance measurements. This software uses the method based on graph reduction. A given distance graph is first recursively decomposed into smaller sub-graphs, each having a unique three-dimensional embedding. Each of these sub-graphs is solved by minimizing the least-square error function. The idea here is to apply a divide-and-conquer approach to find the overall solution of the original distance graph. There are quite a bit of advantages in using this algorithm. Firstly, even if there is insufficient information solving a bigger graph, the method is able to solve uniquely for small chunks of the graph. Secondly, sometimes only the solution to sub-graphs holds importance, not the original graph, so the algorithm can be used to solve only for those important sub-graphs. Third, the algorithm can also determine whether sufficient information is provided to solve the problem. Lastly, the problematic sub-graph i.e., the one which cannot be solved due to erroneous data, can be identified.

3.1.4 Least-Squares Formulation

In this subsection, we will formulate the distance geometry problem as a global least-squares problem. Let us consider the problem with exact distances; the problem can be defined with a set of equality constraints as,

$$|x_i - x_j| = d_{ij}, \quad (i,j) \in S$$

Where S may or may not contain the whole set of distance pairs. If we want to solve these types of problems, we can measure the relative errors between the calculated and given distance using the following equation,

$$\frac{|x_i - x_j|^2 - d_{ij}^2}{d_{ij}^2}, \quad (i,j) \in S$$

This relative error is collected for each pair of points to obtain an overall error function,

$$f(x_1, ...x_n) = \sum_{i,j \in S} [\frac{|x_i - x_j|^2 - d_{ij}^2}{d_{ij}^2}]^2$$

Here we can notice, if the distance constraints are properly met then the error function is equal to zero. Similarly, for problems involving bounds on the distance, we have below inequalities,

$$l_{ij} \leq |x_i - x_j| \leq u_{ij}, \quad (i,j) \in S$$

Then an error function can be written as,

$$f(x_1, ...x_n) = \sum_{i,j \in S} min^2[\frac{|x_i - x_j|^2 - d_{ij}^2}{d_{ij}^2}, 0] + max^2[\frac{|x_i - x_j|^2 - d_{ij}^2}{d_{ij}^2}, 0]$$

It is not very difficult to verify that if all the inequality constraints are met, the error function will be equal to zero.

With the above error function f, we can easily see that a set of coordinates $x_1, x_2, ...x_n$ gives the solution to the distance geometry problem if and only if it is the global minimizer of f and the global minimum should be equal to zero. Therefore, the distance geometry problem can be formulated as an optimization problem.

$$min_{x_1,..,x_n} f(x_1, ..., x_n)$$

More details could be found in [8].

3.1.4.1 DGSOL

DGSOL software package, which was developed by More and Wu [35][36], tries to solve the molecular distance geometry problem by using global smoothing and continuation approach. This particular approach still works without having the availability of all distance or bounds and takes into consideration the least-squares formulation of the distance geometry problem.

The least-squares problem generally consists of many local minimizers. To locate an actual global minimizer, the least-squares function is first transformed into a set of gradually deformed but smoother or easier functions with fewer local minimizers by using the global smoothing and continuation method. This method is used on some small to medium-sized test problems, which consists of approximately 200 points or atoms. From the result, it was evident that the technique was able to find the global minimizer of the least-squares function with a very high probability.

One of the major advantages of using this approach is that it does not require all the distances or bounds to be known. Since the method uses a small number of terms, the cost for solving the distance geometry problem becomes cheap. The technique becomes more practical when only sparse set of distance bounds are available. The bound smoothing technique can be helpful for getting some additional distance data but are generally not so reliable.

3.1.5 Alternating Projection Algorithm

Glunt et al. [17] proposed an Alternating Projection algorithm, which can be used for solving the distance geometry problem with only a given set of bounds on distances. The main idea here is to first determine the set of distances from the given distance bounds. The resulting distance geometry problem is then solved by minimizing an error function(optimization). The program stops when a solution is found; otherwise the violated constraints adjust the distances, and the algorithm is repeated again for a new set of distances.

An important condition for the program to run is the availability of bounds on all the distances. In each iteration, a least-squares problem is solved, and this requires a huge amount of computation. For instance, if Newton's algorithm is used, the total cost can be as high as $O(n)^3$ and if n is too large and the problem needs to be solved in many iterations which can be too expensive to use. Therefore spectral gradient algorithm is used in the alternating projecting algorithm instead, which is much cheaper.

3.2 Crippen and Havel's algorithm

Crippen and Havel [8], pioneering work in the field of distance geometry for molecular conformation, resulted in an algorithm that is used for solving the molecular conformation problem arising in NMR spectroscopy[5] and protein structure determination. There are three main stages involved in the algorithm. The first step is to take the input distance bounds and convert these bounds into distance limits(bound smoothing). The second step is to choose a random value between the limits and fix the distance for all pairs of points. In the final stage, coordinates from the distance constraints(least-squares optimization) are retrieved. A brief description of all the stages is given in the upcoming sections.

3.2.1 Bound Smoothing

Due to imperfect measurements, the distance between the points is generally specified as pairs of upper and lower bounds. In order to calculate the coordinates of the points, the distance bounds should be first tightened into appropriate limits, and the process of converting given bounds into limits is called bound smoothing. Those limits which satisfy triangle inequality are known as triangle inequality limits. A modified version of Floyd's algorithm presented by Dress and Havel [10] is used to convert the bounds into limits to ensure that the limits satisfy the triangle inequality. If there a triangle inequality violation $l_{ij} > u_{ij}$ is found, then the program stops the execution of the current process and repeats itself to find out the limits.

There are some geometric rules which are used in the bound smoothing. For given three points *i,j,* and *k*, let the lower and upper bounds be denoted as l_{ij}, u_{ij}, l_{jk} and u_{jk}. Then the lower and upper limits for the distance between points i and k must agree with the following rules [10],

$$l_{ik} = max(l_{ik}, l_{ij} - u_{jk}, l_{jk} - u_{ij})$$

$$u_{ik} = min(u_{ik}, u_{ij} + u_{jk})$$

Similar to this other rules can also be derived for the distance bounds with more than three points [10].

3.2.2 Metrization

The next step after bound smoothing is to find the distance from these distance limits. This process is called Metrization. As a part of this process, we first take one of the distances and choose a random value between its lower and upper limits as the distance value. Next, we set its lower and upper limits to this random value and recompute the triangle inequality limits using these changed limits as the

upper and lower bounds. Repeating the same process for each distance results in a set of lower and upper triangle inequality limits that are equal to each other and lies within the original limits. The distances which are generated this way form the desired distance matrix that satisfies both the triangle inequality and original limits. For details, please refer to [8],[10].

3.2.3 Embedding

The last piece of the puzzle left in the Crippen and Havel's algorithm is to find the coordinates from the distance matrix. This consists of the following steps:

(i) The distance between each point from the origin is calculated, to avoid overemphasizing any set of points, according to

$$D_{i0}^2 = \frac{1}{N}\sum_{j=1}^{N} D_{ij}^2 + \frac{1}{N^2}\sum_{j=2}^{N}\sum_{k=1}^{j-1} D_{jk}^2$$

where D_{i0} is the distance of the point i from the origin and D_{ij} is the distance between points i and j as shown in the below figure

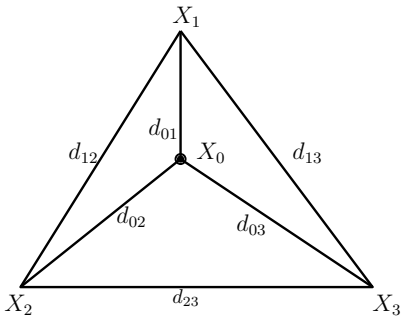


Figure 3.1: Distance between points and origin

(ii) Now the elements a_{ij} of the metric matrix A are computed from the distance of points from the origin as,

$$a_{ij} = \frac{1}{2}(D_{i0}^2 + D_{j0}^2 - D_{ij}^2)$$

(iii) Let W be the diagonal matrix of weights $W = diag(w_1, ..., w_n)$, we assume

all the weights to be 1 in our case the weights, then the B matrix is calculated as,

$$B = W\,A\,W$$

(iv) According to Gale and Householder equation[51], If B matrix is positive semi-definite, the final coordinate matrix X is obtained by diagonalizing the B matrix,

$$B = \sigma\,L^2\,\sigma^{'}$$

and

$$L^2 = [\lambda_1^2, \lambda_2^2, ...\lambda_r^2, 0, ...0]$$

Finally,

$$X = \sigma\,\sqrt{L}$$

where L is the diagonal matrix of latent roots of the B matrix, and σ is the diagonalized eigen vectors of the corresponding latent roots.

Chapter 4

Molecular Distance Geometry approach for the C_α trace problem

Building from the knowledge and material provided in the previous chapters, we will describe the main contribution of this thesis in this chapter. We propose a new method for solving the C_α trace problem using the Molecular Distance Geometry approach. The subsequent sections will discuss the overall proposed methodology, followed by the detailed description of every step of the algorithm. The key advantage of using our approach is that it eliminates the building and searching of a huge protein fragment library.

4.1 Overview of the Method

The method described here consists of several steps to solve the C_α trace problem. An incomplete PDB file that consists of only the coordinates of the C_α atoms is read using the Biopython package[6] developed in python. After reading an incomplete PDB file, the coordinates of the atoms of the single peptide plane are calculated. A peptide plane consists of five atoms, namely two C_α, C, N, and O

atoms. In order to compute the coordinates of these atoms in the peptide plane, we used EMBED algorithm[19], which is used to solve the Molecular Distance Geometry Problem. The computed coordinates are then subjected to appropriate rotation and translation with respect to the first two original C_α coordinates, which are present in the incomplete PDB file. Once, the two C_α atoms are aligned with the two original C_α atoms, the same rotations, and translation are then applied to all the other atoms present in the single peptide plane. Now, the process is repeated for the entire chain of the protein molecule taking two C_α atoms successively, finding the appropriate rotation and translation, and applying the same rotation and translation to all the atoms of the peptide plane. SCWRL4[26] is used to predict the side chains for the protein described above. The above output is then subjected to LBFGS energy minimizer[29], which is implemented using a molecular modeling tool called MESHI[23].

4.2 Proposed Methodology

Our methodology works in four major steps:

a) Step 1: **Prediction of main chain atoms using C_α trace:** This step mainly calculates the coordinate of a single peptide using the prior known bond lengths and bond angles. The coordinates are calculated using the EMBED algorithm[19].

b) Step 2: **Appropriate Rotation and Translation:** This step iteratively calculates the appropriate rotation and translation, which can be applied to all the atoms in the peptide plane based on the two successive C_α atoms in the trace.

c) Step 3: **Side Chain prediction using SCWRL4:** This step uses SCWRL4[26], which is designed for the task of prediction of side-chain conformations given fixed main chain atoms of a protein.

d) Step 4: **Energy Minimization using MESHI:** The final step minimizes the overall energy of the calculated protein structure using the LBFGS

algorithm[29] using a molecular modeling suite MESHI[23].

A flowchart of our proposed approach is given in figure 4.1.

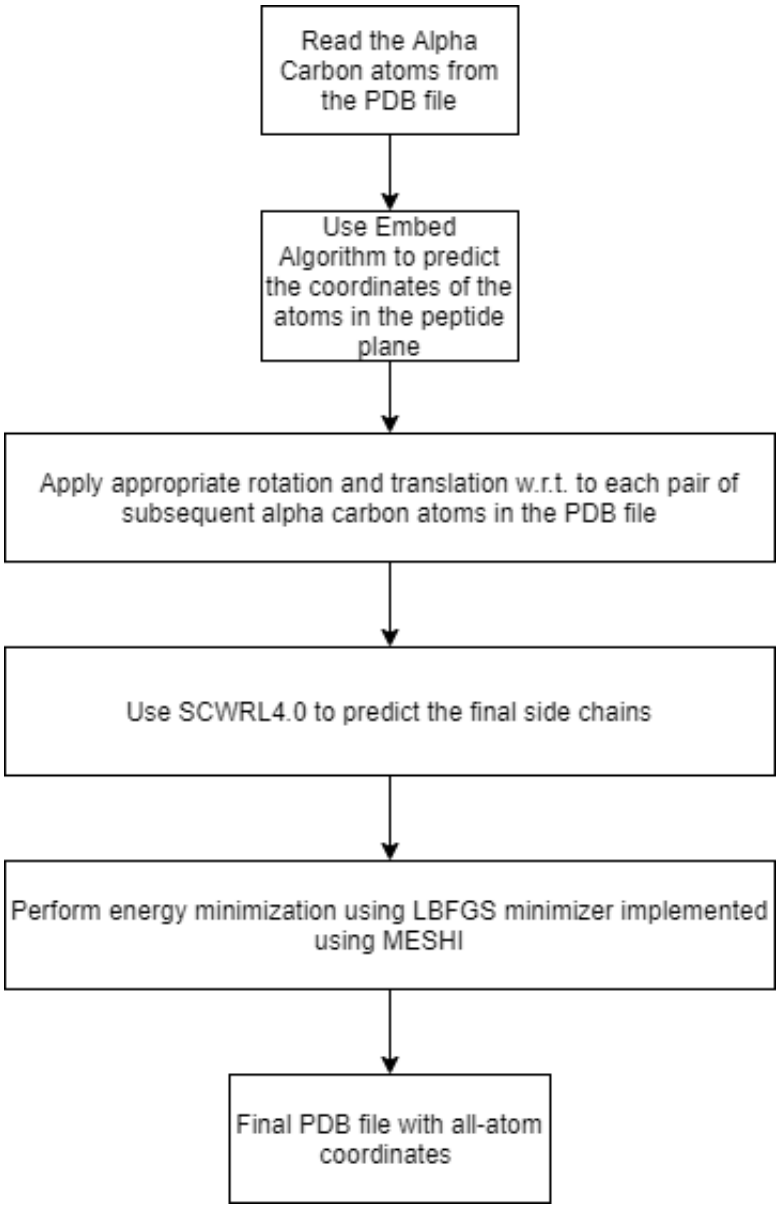


Figure 4.1: Flowchart of our Proposed Methodology

4.2.1 Prediction of main chain atoms using C_α atoms

The main chain of a protein consists of C, N, O atoms in addition to two C_α atoms. A rigid planar structure is formed between these five atoms called the peptide plane, as described in chapter 1 section 1.2.2. Figure 1.3 depicts the

Table 4.1: Distance between each atom in the peptide plane.

	$C_{\alpha i}$	C	O	N	$C_{\alpha i+1}$
$C_{\alpha i}$	0	1.51	2.4	2.42	3.8
C	1.51	0	1.24	1.33	2.43
O	2.4	1.24	0	2.25	2.76
N	2.42	1.33	2.25	0	1.46
$C_{\alpha i+1}$	3.8	2.43	2.76	1.46	0

peptide plane formed by these atoms. It also depicts the various bond lengths and bond angles between different atoms in the peptide plane. The values of covalent bond lengths and angles, as shown in figure 1.3 of a peptide plane, were suggested by Engh and Huber[15]. They were able to find the covalent bond lengths and angles with remarkable accuracy.

Using these values of the bond lengths and bond angles, we have calculated the all remaining the distances between atoms in the peptide plane namely C_α, C, N, O. Since there are a total of five atoms between the peptide plane, we need a total of ten distances to compute the coordinates of the atoms in the peptide plane using EMBED algorithm[19]. We know only four distances between atoms in the peptide plane in terms of bond length. The other six distances are computed by using the cosine law. The distance between two C_α atoms, which is also called as the plane length is taken to be equal to 3.8Å. Thus, all the distances between each pair of atom in the peptide plane are given in table 1.

A version of EMBED algorithm[19] was implemented using python programming language[24]. The author has used the EMBED algorithm to tackle The Point Placement Problem in the inexact model. In this work, the author tries to find the location of n points on a line given only the upper bound and lower bound distances between some pairs of points. The primary motivation of this work comes from the probe location problem in DNA mapping. For solving this problem, the author has developed the DGPL program[24], which takes a set of input distances in the form of upper and lower bounds between the pair of n points and finds the coordinates of the points in the given dimension. The working of the DGPL program is explained below:

DGPL Program

Input data: i. The total number of points used. ii. The embedding dimension.

Output: Final coordinates of the points in the embedding dimension.

Process:

Step 1: A random layout of n given points is created such as $\{p_0, p_1, ..., p_n\}$.

Step 2: If there are unknown distances between certain pairs of points, $[-\infty, \infty]$ is assigned as the values in upper and lower bounds distance matrix.

Step 3: A modified version of the Floyd's shortest path algorithm[19] is used which can convert the given distance bounds into distance limits.

```
procedure Floyd( Natom,Lower,Upper )
       for k from 1 to Natom do
             for i from 1 to Natom - 1 do
                  for j from i + 1 to Natom do
      comment: Path lengths in left-hand network.
                    if Upper[i,j] > Upper[i,k] +  Upper[k,j] then
                             Upper[i,j] :=Upper[i,k] + Upper[k,j];
      comment: Path lengths from left to right-hand network.
                   if Lower[i,j] < Lower[i,k] - Upper[k,j] then
                             Lower[i,j] :=Lower[i,k] - Upper[k,j];
                  else
                   if Lower[i,j] < Lower[j,k] - Upper[k,i] then
                             Lower[i,j] :=Lower[j,k] -  Upper[k,i];
     comment: Check for triangle inequality violations.
                   if Lower[i,j] > Upper[i,j] then
                             exit( ``bad bounds'' );
     endfor endfor endfor
endproc
```

Step 4: A random number is chosen between upper and lower limit. The

process is repeated until all the distances get fixed.

Step 5: B matrix is calculated which is equal to the sum of squared distances between two points and is given by [51]:

$$b_{ij} = (d_{in}^2 + d_{jn}^2 - d_{ij}^2)/2$$

where d_{ij} is the distance between points i and j, n is the starting point(origin) p_0.

Step 6: The eigenvalue decomposition of the B matrix is calculated. The resultant coordinates of the points are found by finding the product of the largest eigenvalue with its corresponding normalized eigenvectors.

We have used the DGPL program[24] explained above to find the coordinates of the atoms of the peptide plane by using only the distances given in Table 1. The coordinates of these atoms were calculated in three-dimensional space.

4.2.2 Appropriate Rotation and Translation

Once the atoms of the single peptide bonds are generated using the EMBED algorithm[19], the atoms of the peptide plane are subjected to appropriate rotations and translation to determine the actual positions of these atoms with respect to the given C_α trace. The process of rotation and translation can be explained by using figure 4.2.

The structure shown in violet color in figure 4.2 depicts the peptide plane calculated using the EMBED algorithm. For doing the translation and rotation, we need to take the first and the last atom of the peptide plane. These atoms are the two C_α atoms. We align these two atoms with the two subsequent C_α atoms in the original trace (shown in blue) by performing appropriate rotation and translation. This process of alignment is then repeated for the entire chain of the protein molecule, every time taking the original two subsequent C_α atoms. The values of rotation and translation calculated at each step are then applied to other atoms (C, N, and O) in the peptide plane. This reconstructs the main chain

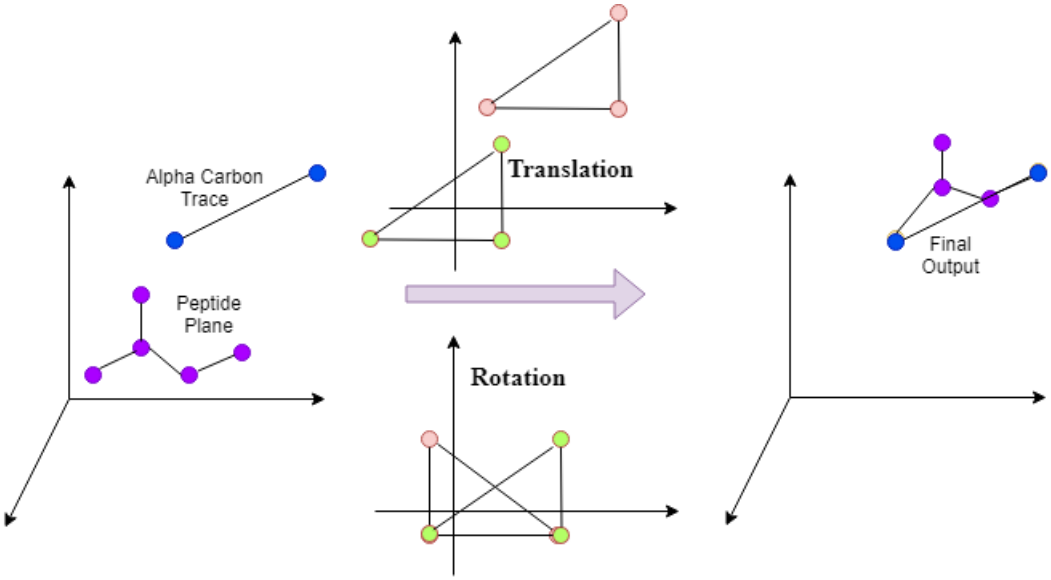


Figure 4.2: Rotation and Translation

of the protein molecule.

We present an algorithm to find appropriate translation and rotation for the atoms in the peptide plane. This algorithm is given below:

Algorithm 1: Rotation and Translation

Result: Final Rotation and Translation
input: EList, CAList
output: finalList
while $i \leftarrow$*length of CAList* **do**
  EListTemp $\leftarrow$ *EList*
  fixedList $\leftarrow$ ($CAList$[i], $CAList$[i+1])
  moveList $\leftarrow$ ($EListTemp$[0], $EListTemp$[4])
  tM $\leftarrow$fixedList[0] - moveList[0]
  fixedList $\leftarrow$ translation(moveList,fixedList,tM)
  θ_1, dir_1 $\leftarrow$ rotation1(moveList,fixedList)
  yList $\leftarrow$ any two random points lying on the Y-axis
  θ_2, dir_2 $\leftarrow$ rotation2(fixedList,yList,moveList)
  θ_3, dir_3 $\leftarrow$ rotation3(moveList,fixedList)
  θ_4, dir_4 $\leftarrow$ rotation4(moveList,fixedList,θ_2,!dir_3)
  while $n \leftarrow$ *length of EListTemp* **do**
    $EListTemp[n] \leftarrow$ rotate(EListTemp[n],
    $\theta_1, \theta_1, \theta_1, \theta_1, dir_1, dir_2, dir_3, dir_4$, tM)
  finalList.append(EListTemp)
return finalList

The above algorithm takes two lists as input, the coordinate list generated by EMBED algorithm (EList), and the given C_α trace atom list (CAList). The

algorithm runs for the length of atoms in the C_α trace. Each time two subsequent C_α atoms are picked from CAList and stored in fixedList. The first and the last element from EList are picked stored in moveList. The goal is to align the first two atoms in moveList with the first two atoms in fixedList. The translation is carried out first, which coincides the first atom in fixedList with the first atom in moveList. Once the two atoms coincide, rotation is carried out next. A total of four rotations are carried out. These sets of rotations need to follow a particular sequence in order to align fixedList and moveList atoms.

In general, the angle between the two vectors X & Y is calculated by using the following formula:

$$\theta = \cos^{-1} \frac{(X.Y)}{\sqrt{X^2 + Y^2}}$$

The direction for rotation i.e., whether to rotate clockwise or anti-clockwise, is determined by finding the determinant (D) between vector X and Y. If D $>$ 0 anti-clockwise rotation otherwise clockwise rotation.

Let us consider vector A as target vector and vector B as reference vector, which are formed by joining atoms in the moveList and fixedList, respectively. Rotation1 is done only on vector A along the z-axis so that both vectors A & B lie in the same plane. Rotation 2 is done along the z-axis on both vectors A & B such that both the vectors lie in the yz-plane. Rotation 3 on vector A such that two vectors get aligned along x-axis. Rotation 4 is done to bring back both the vectors to the original position of vector B along the z-axis. The result of these rotations aligns the two C_α atoms calculated using the EMBED algorithm with the two consecutive C_α atoms from the original trace. The rotations and translation are stored and applied all to other atoms of peptide plane calculated using EMBED algorithm i.e., C, N, and O. The process is then repeated for all the subsequent pairs of C_α atoms until the end of the chain is reached. The resultant list is finally returned, which consists of the coordinates of the atoms of the main chain of a protein molecule.

4.2.3 Side Chain prediction using SCWRL4

Determining side-chain conformations is a vital step in protein structure prediction and protein design. Many side-chain prediction methods are based on sample space, which depends on a rotomar library. A rotomar library is defined as a statistical clustering of side-chains that are observed in known protein structure[11]. There are two types of rotomar libraries, such as backbone-independent[31], where all the side chains are grouped together regardless of the local protein backbone conformation, and backbone-dependent where the frequencies and dihedral angles are varied according to the protein backbone dihedral angles ϕ and ψ[12, 13].

SCWRL3[4] is one of the most used programs for side-chain predictions. As of April 30, 2009, it has around 2986 licenses in 72 countries[26]. It uses a backbone-dependent rotamer library[12], a simple energy function based on the library rotamer frequencies and a purely repulsive steric energy term and uses graph decomposition to solve the combinatorial packing problem[4]. There are three main reasons as to why SCWRL3 became popular. The first one is speed, the second one is accuracy, and the third one is usability. The input to this program is the PDB coordinates for the backbone atoms and outputs the coordinates for the structure with predicted side-chains while maintaining the same residue numbering and chain identifiers as the input structure. One disadvantage of using SCWRL3 is the method used for graph decomposition sometimes does not result in a combinatorial optimization, which can be solved easily and quickly. This may take many hours to finish instead of finishing in seconds.

SCWRL4[26] is a major improvement over SCWRL3. The accuracy of SCWRL4 is greater than the accuracy SCWRL3 or comparable to many other programs that were developed before, for side-chain prediction. Secondly, the speed is greater than SCWRL3 and also maintains the usability. It overcomes the disadvantage of SCWRL3 by ensuring that the program is always able to solve the prediction problem in a reasonable time, even when the graph is not decomposable. This is

achieved by taking an approximation that does not guarantee a global minimum of energy function in a given rotamer search space, but this performs the calculation quickly in most of the cases.

In summary, SCWRL4 is available as a downloadable program which takes input as PDB coordinates calculated from the Step 1 and Step 2 of our approach and predicts the coordinates of side-chain atoms of the protein molecule. This program generates an output PDB file which consists of all the missing atoms along with their respective coordinates.

4.2.4 Energy Minimization using MESHI

MESHI[23] is an object-oriented molecular modeling suite written in Java. The main reason for choosing Java as the language for developing this tool is that Java enforces object-oriented design more vigorously, has a built-in garbage collector, and is platform-independent. Due to this object-oriented approach, every aspect of molecular modeling can be represented by either a class or an interface. In this sense, MESHI consists of classes not just for molecular elements, such as atoms, residues, and proteins, but also for geometrical concepts, such as distances and angles, for energy terms and for algorithmic procedures, such as line-search. The various classes in MESHI are logically grouped together and are arranged in a hierarchy of packages. A brief summary of the five major packages is presented below.

Molecular Elements: This package consists of classes to represent atoms, residues, and proteins. This package also consists of specialized lists. These general-purpose classes are extended by specific molecular models such as All-atom and C_α-only proteins in the form of sub-packages.

Geometry: This package consists of classes that represent coordinates, distances, angles, and torsion angles, as well as specialized containers. The objects of these classes can be shared among different energy functions.

Energy: The classes included in this package consists of abstract classes that

represent different aspects of energy terms. These abstract classes perform activities such as reading parameters from files, binding of atoms and geometry elements to their different roles in the energy function, and the actual evaluation of the energy. The Total energy class is a container which is able to store a large number of energy terms.

Optimizers: This package consists of classes that implement optimization and conformational search algorithms. The most useful algorithms which rely on energy function derivability such as LBFGS[29] and MCM[28] are implemented in this package.

Util: The classes included in this package are able to handle files, lists, and command interpretation.

We used the MESHI library and implemented a Java Class, which is used to minimize the energy of the protein using the LBFGS algorithm[29]. This class minimizes the protein structure according to standard energy terms, such as a bond, angle, plane, out-of-plane (chirality), torsion pair (Ramachandran+rotamers). The output of the program is the final PDB file, which contains a complete protein structure.

4.3 Experminental results

We have implemented our proposed approach for solving the C_α trace problem using python 3.7 on a computer with the following configurations: Intel(R) Core(TM) i5-3230M CPU @ 2.60GHz, OS: Windows 10, Architecture: 64-bit. We have used *Click* python package to integrate different parts of our approach. Using this python package, we are able to streamline each step in our approach using a single *Command Line Interface (CLI)* application. In this section, we will focus on presenting various computational results we have obtained using the approach mentioned in section 4.2. This will be followed by a discussion on these results and conclusions that can be derived from these results. The following chapter will

discuss the final conclusions and scope of future work with the current approach.

We have experimented with proteins whose C_α trace is given in the Protein Data Bank (PDB)[2]. We implemented our approach, as described in section 4.2, in order to determine the complete full 3-D structure of the protein molecule. Also, we have used other state of the art methods such as PULCHRA[45] (discussed in section 2.1), BBQ[18] (discussed in section 2.2) and PD2Main[34] (discussed in section 2.3) for comparing our results. All these methods are used to solve the C_α trace problem, but these methods use a large protein fragment library in order to calculate the main chain atoms of the protein molecule.

We have used two different approaches to evaluate the quality of our approach in comparison to other approaches. We have also made a Run-time comparison between different methods to gain further insight into our approach.

The first method to evaluate our approach with other methods is the root-mean-square deviation of atomic positions (or simply root-mean-square deviation, RMSD). RMSD is the measure of the average distance between the atoms (usually the backbone atoms) of two superimposed protein molecules. It is given by:

$$RMSD = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\delta_i}$$

where δi is the distance between atom i of the target structure and atom i of the reference structure. The RMSD is often calculated for the heavy backbone atoms such as C, N, O, and C_α.

The next approach to evaluate our approach is the measure the stereo-chemical quality of our protein structure. For calculating the stereo-chemical quality, we have used PROCHECK[27], which computes the Ramachandran Plot[42] of the predicted structure. A Ramachandran plot is a good way to visualize energetically allowed regions in a protein structure. It plots dihedral angles ϕ against ψ of amino acid residues in a protein structure. This plot of dihedral angles is a good way to show the distribution of amino-acid residue in a single protein structure. This can be used for structure validation. A large percentage of data points in the favored regions of the plot implies a better quality structure.

In order to evaluate the results of our approach, two kinds of PDB files will be used. First, the actual C_α trace files PDB files from the Protein Data Bank. These files do not have all the coordinates of the atoms in the protein chain. Second, the synthetic C_α trace files which are made by deleting all the other atoms except C_α atoms. An RMSD value comparison is made between our method and the other methods such as PULCHRA, BBQ, and PD2 ca2main in terms of only the backbone atoms, namely C, O, N, and C_α. The other comparison is in terms of the Ramachandran Plot. Here, a Ramachandran plot will be plotted for each structure predicted through different methods, and the percentage of residues in the favored regions will be compared.

PDB ID	**Residues**	**PULCHRA**[45]	**BBQ**[18]	**PD2ca2Main**[34]
1HIO	95	1.339	1.698	1.310
1A1D	146	1.383	1.695	1.340
1F6G	160	1.272	1.63	1.322
1BDX	190	1.413	1.746	1.338
1AE4	324	1.304	1.624	1.315
1LBG	357	1.359	1.641	1.323
2BK1	444	1.446	1.838	1.405
1QCR	446	1.355	1.641	1.330
2BK2	456	1.316	1.650	1.337
1KVP	497	1.322	1.627	1.328

Table 4.2: RMSD comparison on incomplete PDB files relative to our method.

Table 4.2. shows RMSD comparison on the incomplete PDB files between our method and other methods such as PULCHRA, BBQ, and PD2 ca2main. We can note that the backbone coordinates calculated by our method on the basis of the C_α trace is comparable to other methods. This is evident from the low RMSD values between the structure computed by our method and the structure computed by other methods.

Table 4.3. on the other hand, measures the RMSD value between our method and actual protein structure. Table 4.3 also captures the RMSD values for other methods such as PULCHRA, BBQ, and PD2Ca2Main. The synthetic C_α trace files are generated by deleting all the atoms except C_α from the actual PDB file of

PDB ID	Residues	PULCHRA[45]	BBQ[18]	PD2ca2Main[34]	Our Approach
1AHL	49	0.9310	1.3988	0.5686	1.3158
4PTI	58	0.6034	1.1553	0.4163	1.3277
1AIW	62	0.8758	1.2522	0.7247	1.2522
1CTF	68	0.5174	1.1947	0.2126	1.3308
1ADR	76	0.5427	1.1770	0.4140	1.3636
1UBQ	76	0.4091	1.1391	0.3363	1.4073
1PLC	99	0.5921	1.1398	0.4107	1.3798
2MHR	118	0.3973	1.1118	0.2617	1.3840
2LYM	129	0.5201	1.14779	0.3375	1.3749
1A3K	137	0.4715	1.1112	0.2426	1.3517
111M	154	0.3834	1.0880	0.2925	1.3320
1AEW	170	0.3517	1.1045	0.2845	1.3205
1VCA	199	0.5846	1.1423	0.3526	1.2658
1TIM	247	0.6356	1.1881	0.5796	1.3962
1A3X	487	0.6118	1.1493	0.4706	1.3456
1A5U	519	0.4825	1.0776	0.3125	1.3411
1A49	519	0.4622	1.0869	0.3193	1.3369

Table 4.3: RMSD comparison on synthetic PDB files between different methods and actual structure.

the protein molecule. A graph is plotted between the number of residues on the x-axis and the RMSD values obtained from different approaches on the y-axis, as shown in figure 4.2. Moreover, we can see that the RMSD values lie between 1.2 and 1.4 as the number of residues varies. More importantly, our method shows RMSD values, which are very comparable to the BBQ approach. Thus, from table 4.3 and figure 4.3, it is evident that our method shows RMSD values, which are comparable to other methods.

PDB ID	Residues	PULCHRA[45]	BBQ[18]	PD2ca2Main[34]	Our Approach
1HIO	95	87.7%	91.7%	94.3%	77.8%
1A1D	146	66.2%	69.2%	77.6%	69.2%
1F6G	160	82%	84.5%	89.5%	75.5
1BDX	190	85.9%	89.7%	92.4%	75.3%
1AE4	324	78.4%	82.9%	88.2%	70%
1LBG	357	74.1%	72.8%	80.1%	75%
2BK1	444	77.3%	81.3%	84.1%	73.3%
1QCR	446	78.7%	77.6%	81.4%	73.6%
2BK2	456	81.1%	85.9%	90.2%	70.5%
1KVP	497	78.2%	80.1%	87.3%	68.9%

Table 4.4: Ramachandran Plot allowed region comparison on incomplete PDB files between different methods.

Table 4.4 shows the percentage of total residues in a protein molecule that lies in the favorable region in the Ramachandran plot. The PDB files used here

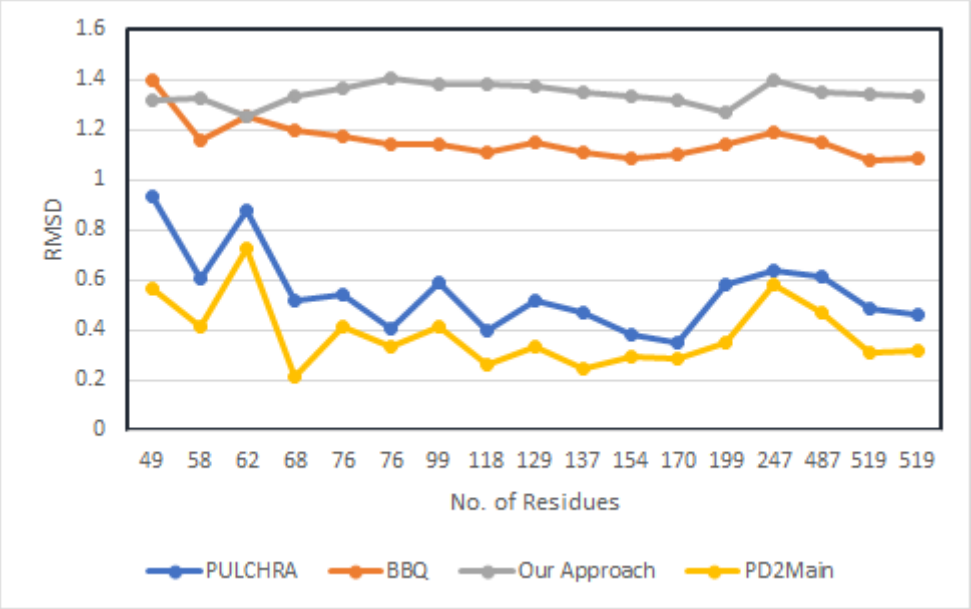
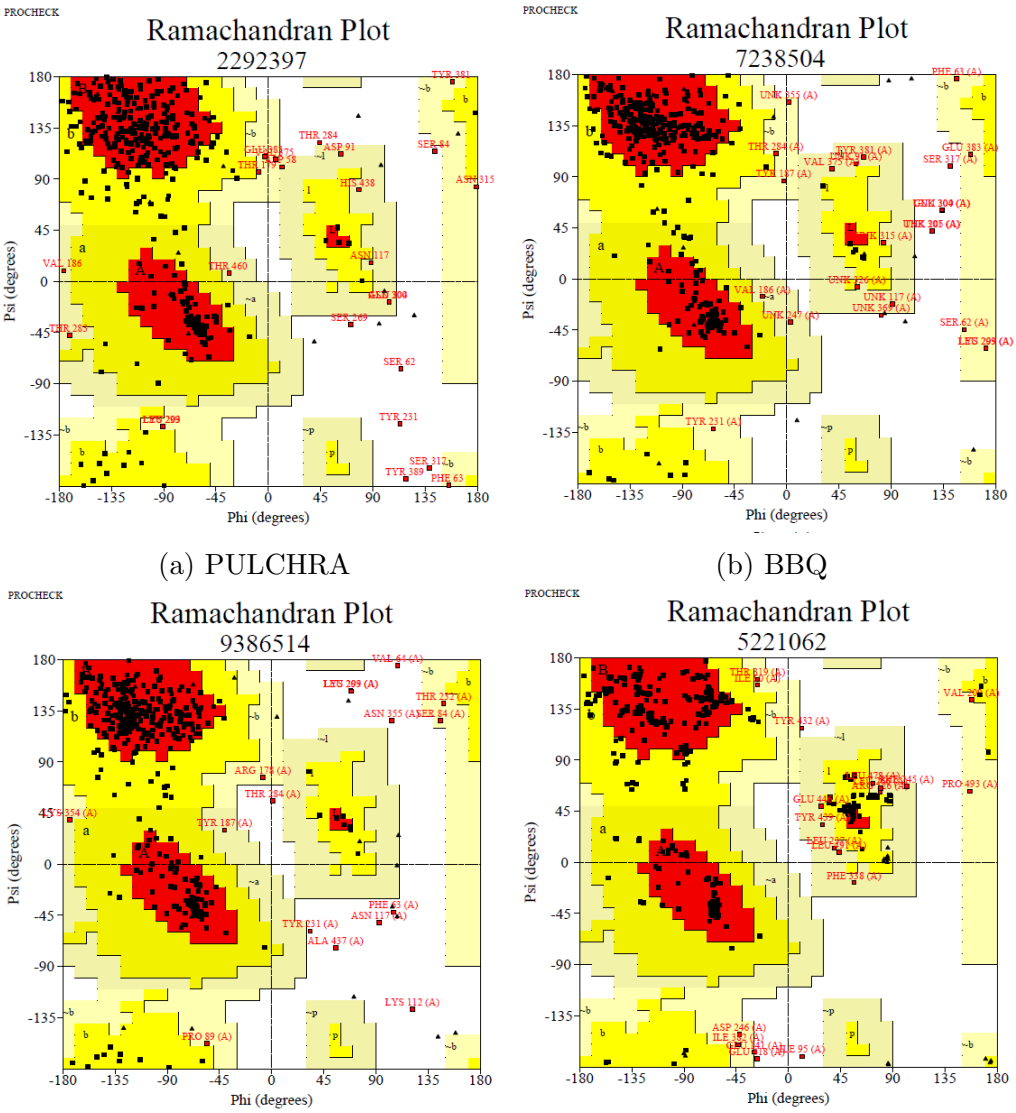


Figure 4.3: Graph depicting RMSD vs No. of Residues for different methods

are incomplete in the Protein Data Bank. The values obtained using different approaches are captured and compared in table 4.4. Figure 4.4 depicts actual Ramachandran plots for one of the PDB files with PDB ID-2BK1. From figure 4.4, we can note that the area described in red represents the favored region of the Ramachandran plot. Each black dot represents a residue present in the given protein molecule. The higher is the percentage of residues falling in the favorable regions of the plot, the higher is the stereo-chemical quality of the protein. The figure also depicts the Ramachandran Plot generated using all the four techniques. We can see that using our approach, more than 70% of the residues are falling under the favorable region, which shows high stereo-chemical quality.

Table 4.5 shows the percentage of residues lying in the allowed region of the Ramachandran plot for synthetic PDB files for different methods. Figure 4.5 is a graphical representation of table 4.5, where the x-axis represents the number of residues in a protein molecule, and the y-axis represents the percentage of residues lying in the allowed regions. From figure 4.5, it is clearly evident that using our approach; we can get 75% of the residues lying in favorable regions most of the time. This is true for protein molecules having a large number of residues, greater than 154.



(a) PULCHRA (b) BBQ

(c) PD2ca2Main (d) Our Approach

Figure 4.4: A Ramachandran plot generated for 2BK1 by PROCHECK[27]

Figure 4.6 depicts an actual Ramachandran plot for one of the PDB files with PDB ID-5PCA. We can see Ramachandran Plots generated for a PDB file after adding the coordinates of the missing atoms using all the different techniques. This PDB file has 307 total number of residues out, which 75.7% of the residues lie in the allowed region of the Ramachandran plot using our approach. Also, 5PCA is an example of a synthetic PDB file.

Thus, from all the results that we have seen so far, we can safely conclude that our approach of using the Molecular Distance Geometry technique to solve C_α trace problem is comparable to other methods. This can be seen from the RMSD

PDB ID	Residues	PULCHRA[45]	BBQ[18]	PD2ca2Main[34]	Our Approach
1AHL	49	69.4%	77.8%	80.5%	80.6%
4PTI	58	82.6%	91.3%	91.3%	80.4%
1AIW	62	48%	68%	62%	68%
1CTF	68	86.4%	88.1%	94.9%	72.9%
1ADR	76	83.1%	82.1%	91%	68.7%
1UBQ	76	84.8%	92.4%	95.5%	59.1%
1PLC	99	84.1%	89%	90.2%	65.9%
2MHR	118	41.9%	88.7%	94.3%	70.8%
2LYM	129	77.9%	86.7%	92%	74.3%
1A3K	137	79.7%	85.6%	87.3%	64.4%
111M	154	89.1%	94.2%	92%	75.2%
1AEW	170	89%	92.9%	94.8%	75.3%
1VCA	199	80.09%	81.8%	88.1%	75%
1TIM	247	79.5%	82.9%	85.8%	75.7%
1A3X	487	80.4%	84.6%	86.7%	73.7%
1A5U	519	81.5%	87.6%	91.3%	76.5%
1A49	519	83.2%	86.5%	92.2%	75.4%

Table 4.5: Ramachandran Plot allowed region comparison on synthetic PDB files between different methods.

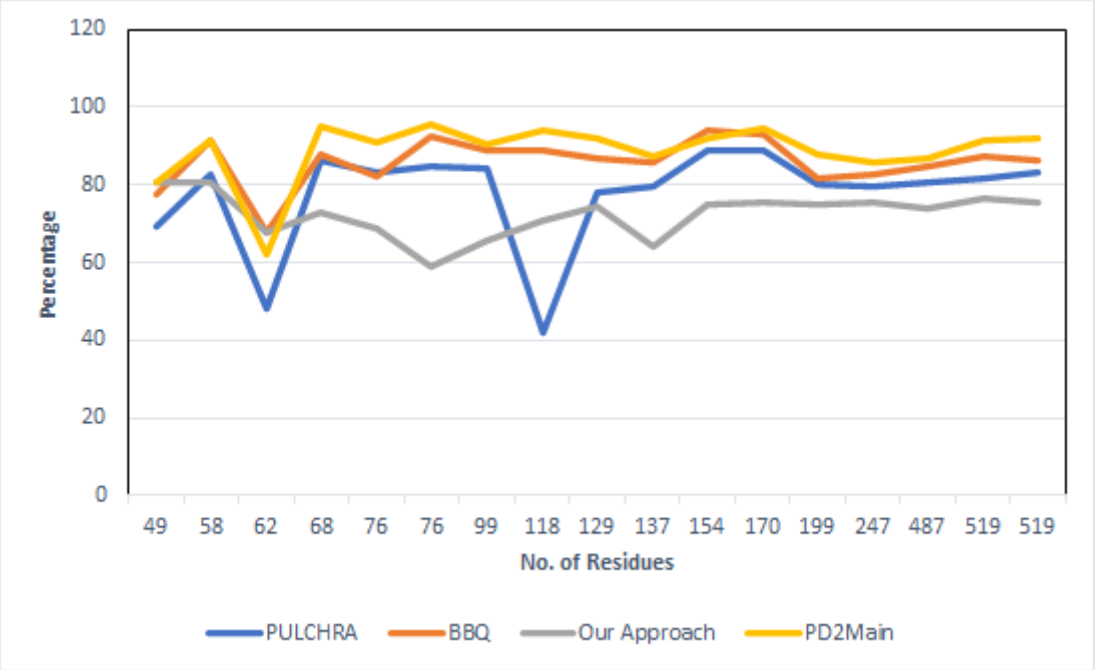
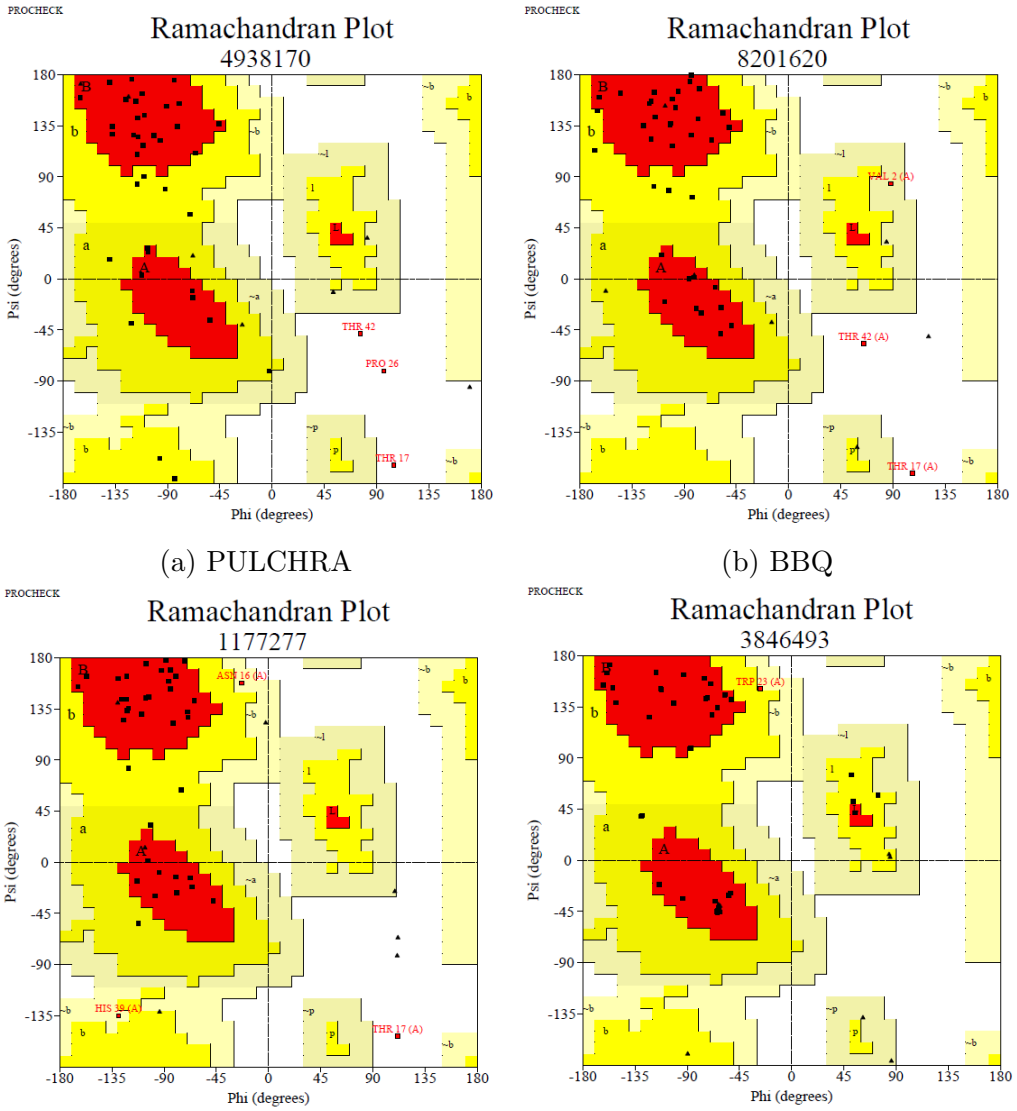


Figure 4.5: Graph depicting Allowed Region Percentage vs No. of Residues for different methods

values and Ramachandran plot percentage values. The novelty of our work lies in the fact that our approach only uses distances to find the coordinates of the missing atoms in the peptide plane.

Now, we are going to present a runtime comparison of our approach with other methods such as BBQ and PULCHRA. We are not able to calculate runtime for



(a) PULCHRA (b) BBQ

(c) PD2ca2Main (d) Our Approach

Figure 4.6: A Ramachandran plot generated for 1AHL by PROCHECK[27]

PD2Ca2Main because it is available as a web application. We were able to run BBQ and PULCHRA in our local machine. Thus, the configuration under which BBQ, PULCHRA, and our approach solved the problem is the same.

Table 4.6 shows the runtime comparisons between different methods for solving the C_α trace problem. The time is measured in milliseconds for all three approaches, and a graph is also plotted, as shown in figure 4.6. The graph shows a number of residues in a protein molecule (x-axis) against running time in milliseconds (y-axis). It is clearly evident that the run time of our approach is significantly less when compared to other approaches. We notice that the highest run time is

PDB ID	No. of Residues	PULCHRA[45]	BBQ[18]	Our Approach
1AHL	49	61.22	1054.65	30.97
4PTI	58	63.907	1078.88	43.97
1AIW	62	66.26	1186.15	36.97
1CTF	68	66.91	1204.72	39.97
1ADR	76	97.93	1276.143	43.97
1UBQ	76	83.9041	1261.53	43.97
1PLC	99	92.90	1189.49	55.96
1ACX	108	289.22	1320.49	58.96
2MHR	118	120.7296	1187.52	64.95
2LYM	129	184.6233	1233.72	66.96
1A3K	137	118.185	1040.407	73.95
111M	154	130.01	1296.3922	79.949
1AEW	170	144.417	1221.387	155.902
1VCA	199	159.9616	1248.785	98.93
1TIM	247	270.248	1236.4231	122.924
1AE4	324	447.5575	1283.888	182.885
1A3X	487	426.142	1359.248	233.854
1A5U	519	415.66	1337.627	244.851
1A49	519	420.906	1405.1902	252.842

Table 4.6: Run time comparisons between different methods(in milliseconds).

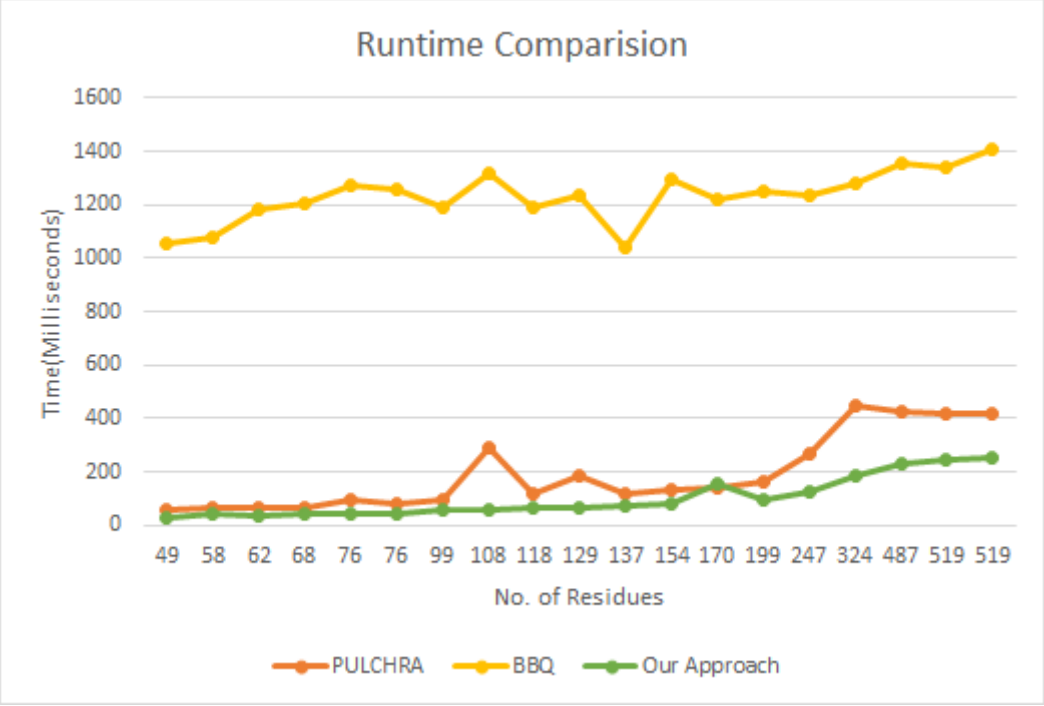


Figure 4.7: Graph depicting Run time comparison between different methods.

shown by the BBQ method in all cases. We can see a linear increase in the run time using our approach with an increasing number of residues.

If we analyze the data given in table 4.6, we can say that, on average, our approach shows a 44% run time improvement as compared to PULCHRA. Similarly, when compared to BBQ, the run time improvement is nearly 92% using our approach. The reason for this improved run time is because of the fact that we are not searching the protein fragment library as opposed to other approaches that are mentioned. The searching of the protein fragment library adds to the overall complexity of the algorithm. Moreover, building a protein fragment library is requires a lot of effort and domain knowledge.

The primary reason for getting higher RMSD values and a low percentage of residues in the favored regions of the Ramachandran plot as compared to other methods is because we are using average values of bond lengths and bond angles. Now, these average values can sometimes deviate from the real data. Since we are calculating the coordinates of the atoms using these averages distances, we get a structure that deviates from the actual structure. Moreover, the protein molecules which are found in Protein Data Bank sometimes go through different optimization procedures, which further optimizes the resultant geometry of the structure determined using crystallographic experiments. This means that different optimization procedures can sometimes produce, to some degree, a different set of coordinates for a given structure. The fragment library search methods have an intrinsic edge in this regard.

These reasons do not cause our approach to lose any kind of importance. The major advantage of our approach is the faster run time as compared to other methods. Moreover, our approach is very flexible because it uses geometrical constraints and is independent of using any real protein structures. This makes our approach ideal for solving non-complete models of a protein molecule that are present in theory.

Chapter 5

Conclusions

This thesis contributes towards the goal of providing an efficient solution to the C_α trace problem using only the set of distances. C_α trace problem is one of the critical problems in the field of protein structure determination. Many solutions are available for solving this problem. Almost all the solutions use some form of protein fragment library search. Building this protein fragment library requires a lot of effort and domain knowledge. Furthermore, the process becomes complex to search for a matching fragment from the vast fragment library. Here in this thesis, we present a simple approach that does not involve any kind of protein fragment library for predicting the backbone atoms of the protein molecule. It simply uses one of the Molecular Distance Geometric Approach to find the main chain atoms of the protein molecule.

Chapter 1 introduced the problem statement in detail, along with all the preliminary definitions and concepts required to understand the problem statement clearly.

Chapter 2 discussed the prior work done in the field of protein structure prediction using only C_α trace. We can see that all these methods are based upon the fundamental concept of using a protein fragment library in order to predict the structure of the unknown protein molecule. Both PULCHRA[45] and BBQ[18] are available as downloadable softwares and can be used locally. Both these

techniques use a large protein fragment library built using Protein Data Bank. PD2Ca2Main[34] is available as a web application. This technique is based on using a Structural Alphabet(SA). A Structural Alphabet is constructed by performing a Gaussian Mixture Modelling technique on a large protein fragment library. This resulted in a selection of 528 components. Moreover, an analytical technique for calculating the coordinates of the main chain of a protein molecule, is also presented. A detailed description of these techniques was discussed.

Chapter 3 discussed in detail various distance geometry techniques and tools which can be used to solve the Molecular Distance Geometry Problem. EMBED algorithm, proposed by Crippen and Havel[8], is a very fundamental technique that embeds points in space, is also explained in detail. This algorithm works in three stages. This algorithm forms the basis of our proposed approach.

The main contribution of this thesis comes in chapter 4, which explains in detail our proposed approach for solving C_α trace problem using molecular distance geometry technique. All the major four steps involved in solving the problem are explained in detail. This chapter also consists of experimental results performed, which compares our approach to other methods used for solving the same problem. The other methods which are presented use a large protein fragment library to build a complete protein molecule. It requires a lot of effort and domain knowledge to construct these fragment libraries. Our approach, on the other, does not involve the use of any such fragment library and relies upon distance geometry techniques to build a complete protein molecule.

From the different results we obtained by running our algorithm against PDB files consisting of a different number of residues, we can conclude that our approach is comparable to other methods in terms of Root Mean Squared Deviation (RMSD) and Ramachandran Plot. We can also conclude that our approach is simpler to implement and also takes less run time when compared to other methods. All these results can be verified by various tables and graphs that are being plotted in section 4.3.

5.1 Future work

There are several areas where further work can be done. The current work can be further extended to take *cis* and *trans* configuration of a protein molecule into account. A *cis* configuration is defined when both C_α atoms in a single peptide lies on the same side of the peptide bond, whereas in *trans* configuration both C_α atoms lies on the different side of the peptide plane. A peptide bond is a bond between carbon and nitrogen atom in the peptide plane.

Another direction to further extend the work will be to use the Distance Matrix Completion Algorithm (DMCA)[41]. In this case, we can use four subsequent C_α atoms and use DMCA to find the missing distances between all the atoms between these four subsequent C_α atoms. These distances can then be used to find the coordinates of the missing atoms.

A potential area will be to explore whether these distance geometry approaches can be used for determining the side-chain atoms of the protein molecule. Moreover, we can have an hybrid approach between distance geometry approach and the database approach where the database can be used to obtain the average values for bond lengths and bond angles for a peptide plane. These average values should be obtained before using the distance geometry approach.

BIBLIOGRAPHY

[1] Beta carbon. Library Catalog: foldit.fandom.com.

[2] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, TN Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

[3] Frances C Bernstein, Thomas F Koetzle, Graheme JB Williams, Edgar F Meyer, Michael D Brice, John R Rodgers, Olga Kennard, Takehiko Shimanouchi, and Mitsuo Tasumi. The protein data bank: a computer-based archival file for macromolecular structures. *Archives of biochemistry and biophysics*, 185(2):584–591, 1978.

[4] Adrian A Canutescu, Andrew A Shelenkov, and Roland L Dunbrack Jr. A graph-theory algorithm for rapid protein side-chain prediction. *Protein science*, 12(9):2001–2014, 2003.

[5] John Cavanagh, Wayne J Fairbrother, Arthur G Palmer III, and Nicholas J Skelton. *Protein NMR spectroscopy: principles and practice*. Elsevier, 1995.

[6] Peter J. A. Cock, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, and Michiel J. L. de Hoon. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, 03 2009.

[7] Wikimedia Commons. File:alpha-amino-acid-general-2d.png — wikimedia commons, the free media repository, 2015. [Online; accessed 17-February-2020].

[8] Gordon M Crippen, Timothy F Havel, et al. *Distance geometry and molecular conformation*, volume 74. Research Studies Press Somerset, England, 1988.

[9] Jan Drenth. *Principles of protein X-ray crystallography*. Springer Science & Business Media, 2007.

[10] Andreas WM Dress and Timothy F Havel. Shortest-path problems and molecular conformation. *Discrete Applied Mathematics*, 19(1):129–144, 1988.

[11] Roland L Dunbrack Jr. Rotamer libraries in the 21st century. *Current opinion in structural biology*, 12(4):431–440, 2002.

[12] Roland L Dunbrack Jr and Fred E Cohen. Bayesian statistical analysis of protein side-chain rotamer preferences. *Protein Science*, 6(8):1661–1681, 1997.

[13] Roland L Dunbrack Jr and Martin Karplus. Backbone-dependent rotamer library for proteins application to side-chain prediction. *Journal of molecular biology*, 230(2):543–574, 1993.

[14] Ioannis Z Emiris and Ioannis Psarros. Counting euclidean embeddings of rigid graphs. *arXiv preprint arXiv:1402.1484*, 2014.

[15] Richard A Engh and Robert Huber. Accurate bond and angle parameters for x-ray protein structure refinement. *Acta Crystallographica Section A: Foundations of Crystallography*, 47(4):392–400, 1991.

[16] Janice Glasgow, Tony Kuo, and Jim Davies. Protein structure from contact maps: A case-based reasoning approach. *Information Systems Frontiers*, 8(1):29–36, 2006.

[17] W Glunt, TL Hayden, and M Raydan. Molecular conformations from distance matrices. *Journal of Computational Chemistry*, 14(1):114–120, 1993.

[18] Dominik Gront, Sebastian Kmiecik, and Andrzej Kolinski. Backbone building from quadrilaterals: a fast and accurate algorithm for protein backbone reconstruction from alpha carbon coordinates. *Journal of computational chemistry*, 28(9):1593–1597, 2007.

[19] Timothy F Havel. Distance geometry: Theory, algorithms, and chemical applications. *Encyclopedia of Computational Chemistry*, 120:723–742, 1998.

[20] B.A. Hendrickson. *The Molecule Problem: Determining Conformation from Pairwise Distances*. PhD thesis, Cornell University, 1990.

[21] GYassineMrabetTalkThis W3C-unspecified vector image was created with Inkscape. English: Peptide bond formation.Français : Formation d'une liaison peptidique.Polski: Kondensacja aminokwasów – tworzenie wiazania peptydowego.: .Català: Formació de l'enllaç peptídic, August 2007.

[22] Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, 22(12):2577–2637, 1983.

[23] Nir Kalisman, Ami Levi, Tetyana Maximova, Dan Reshef, Sharon Zafriri-Lynn, Yan Gleyzer, and Chen Keasar. Meshi: a new library of java classes for molecular modeling. *Bioinformatics*, 21(20):3931–3932, 2005.

[24] Varaharajan Kannan Kishore Kumar. The point placement problem in an inexact model and its applications. 2015.

[25] Sebastian Kmiecik, Dominik Gront, Michal Kolinski, Lukasz Wieteska, Aleksandra Elzbieta Dawid, and Andrzej Kolinski. Coarse-grained protein models and their applications. *Chemical reviews*, 116(14):7898–7936, 2016.

[26] Georgii G Krivov, Maxim V Shapovalov, and Roland L Dunbrack Jr. Improved prediction of protein side-chain conformations with scwrl4. *Proteins: Structure, Function, and Bioinformatics*, 77(4):778–795, 2009.

[27] Roman A Laskowski, Malcolm W MacArthur, David S Moss, and Janet M Thornton. Procheck: a program to check the stereochemical quality of protein structures. *Journal of applied crystallography*, 26(2):283–291, 1993.

[28] Zhenqin Li and Harold A Scheraga. Monte carlo-minimization approach to the multiple-minima problem in protein folding. *Proceedings of the National Academy of Sciences*, 84(19):6611–6615, 1987.

[29] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[30] Pu Liu, Dimitris K Agrafiotis, and Douglas L Theobald. Fast determination of the optimal rotational matrix for macromolecular superpositions. *Journal of computational chemistry*, 31(7):1561–1563, 2010.

[31] Simon C Lovell, J Michael Word, Jane S Richardson, and David C Richardson. The penultimate rotamer library. *Proteins: Structure, Function, and Bioinformatics*, 40(3):389–408, 2000.

[32] James T MacDonald, Katarzyna Maksimiak, Michael I Sadowski, and William R Taylor. De novo backbone scaffolds for protein design. *Proteins: Structure, Function, and Bioinformatics*, 78(5):1311–1325, 2010.

[33] Mariusz Milik, Andrzej Kolinski, and Jeffrey Skolnick. Algorithm for rapid reconstruction of protein backbone from alpha carbon coordinates. *Journal of Computational Chemistry*, 18(1):80–85, 1997.

[34] Benjamin L Moore, Lawrence A Kelley, James Barber, James W Murray, and James T MacDonald. High–quality protein backbone reconstruction from alpha carbons using gaussian mixture models. *Journal of computational chemistry*, 34(22):1881–1889, 2013.

[35] Jorge J Moré and Zhijun Wu. Global continuation for distance geometry problems. *SIAM Journal on Optimization*, 7(3):814–836, 1997.

[36] Wu Zhijun More, Jorge. Distance geometry optimization for protein structures. *Journal of Global Optimization*, 15(3):219–234, 1999.

[37] Bernard Offmann, Manoj Tyagi, and Akexandre G de Brevern. Local protein structures. *Current Bioinformatics*, 2(3):165–202, 2007.

[38] Alessandro Pandini, Arianna Fornili, and Jens Kleinjung. Structural alphabets derived from attractors in conformational space. *BMC bioinformatics*, 11(1):97, 2010.

[39] Britt H Park and Michael Levitt. The complexity and accuracy of discrete state models of protein structure. *Journal of molecular biology*, 249(2):493–507, 1995.

[40] Linus Pauling, Robert B Corey, and Herman R Branson. The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain. *Proceedings of the National Academy of Sciences*, 37(4):205–211, 1951.

[41] Md Zamilur Rahman, Udayamoorthy Navaneetha Krishnan, Cory Jeane, Asish Mukhopadhyay, and Yash P Aneja. A distance matrix completion approach to 1-round algorithms for point placement in the plane. In *Transactions on Computational Science XXXIII*, pages 97–114. Springer, 2018.

[42] Gopalasamudram Narayana Ramachandran. Stereochemistry of polypeptide chain configurations. *J. Mol. Biol.*, 7:95–99, 1963.

[43] Antonio Rey and Jeffrey Skolnick. Efficient algorithm for the reconstruction of a protein backbone from the α-carbon coordinates. *Journal of computational chemistry*, 13(4):443–456, 1992.

[44] Marianne J Rooman, Joaquin Rodriguez, and Shoshana J Wodak. Automatic definition of recurrent local structure motifs in proteins. *Journal of Molecular Biology*, 213(2):327–336, 1990.

[45] Piotr Rotkiewicz and Jeffrey Skolnick. Fast procedure for reconstruction of full-atom protein models from reduced representations. *Journal of computational chemistry*, 29(9):1460–1465, 2008.

[46] James B Saxe. Embeddability of weighted graphs in k-space is strongly np-hard. In *Proc. of 17th Allerton Conference in Communications, Control and Computing, Monticello, IL*, pages 480–489, 1979.

[47] Manfred J Sippl and Harold A Scheraga. Cayley-menger coordinates. *Proceedings of the National Academy of Sciences*, 83(8):2283–2287, 1986.

[48] Douglas L Theobald. Rapid calculation of rmsds using a quaternion-based characteristic polynomial. *Acta Crystallographica Section A: Foundations of Crystallography*, 61(4):478–480, 2005.

[49] Wikipedia. Distance geometry — wikipedia, the free encyclopedia, 2020. [Online; accessed 10-February-2020].

[50] Jeong-Mi Yoon, Yash Gad, and Zhijun Wu. Mathematical modeling of protein structure using distance geometry. *Department of Computational & Applied Mathematics, Rice University*, 2000.

[51] Gale Young and Alston S Householder. Discussion of a set of points in terms of their mutual distances. *Psychometrika*, 3(1):19–22, 1938.

VITA AUCTORIS

Lokesh Gupta was born in Faridabad, India in the year 1992. He passed Bachelor of Technology in Computer Engineering from Maharishi Markandeshwar University, India in the year 2014. He is currently a candidate for the Masters degree in Computer Science at the University of Windsor, Ontario and to graduate in Winter 2020.