Master's thesis
Theoretical Physics

Large scale atomistic simulations of complex IV reveal novel protein-lipid interactions

Aapo Malkamäki
May 21, 2018

Supervisor: Vivek Sharma
Examiners: Ilpo Vattulainen

UNIVERSITY OF HELSINKI
DEPARTMENT OF PHYSICS

P.O. Box 64 (Gustaf Hällströmin katu 2a)
FI-00014 University of Helsinki

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta/Osasto — Fakultet/Sektion — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Physics

Tekijä — Författare — Author
Aapo Malkamäki

Työn nimi — Arbetets titel — Title

Large scale atomistic simulations of complex IV reveal novel protein-lipid interactions

Oppiaine — Läroämne — Subject
Theoretical Physics

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis	May 21, 2018	52

Tiivistelmä — Referat — Abstract

In mitochondria and many bacteria, the electron transport chain produces energy from foodstuff as a part of cell respiration. Complex IV, also known as Cytochrome *c* Oxidase, is the last protein complex in the electron transport chain. It couples electron transport with the transfer of protons across the inner mitochondrial membrane (or the cell membrane in bacteria). The pumped protons produce a proton-motive force, which drives adenosine triphosphate synthase to generate adenosine triphosphate molecules used as energy currency in many cellular functions. Dysfunction of complex IV may cause myopathies and other mitochondrial malfunctions, and therefore it is important to understand how this enzyme functions and is regulated.

The work performed in this Thesis provides novel insights into the intricate function of the enzyme and reveals the importance of lipid-protein interactions that turn out to be critical in the enzyme function. These insights provide new ways to better understand how cardiolipin as a key lipid in mitochondrial membranes participates in proton uptake pathways, and whether cardiolipin also has an important role in complex IV dimerization.

Six large-scale atomistic molecular dynamics simulations of complex IV were performed, including simulations of the dimeric as well as the monomeric complex IV. In each simulation, the membrane consisted of three kinds of primary lipids found in the inner mitochondrial membrane. All the simulations were two to three microseconds long, therefore representing the current state-of-the-art in membrane-protein simulations in this context.

The simulation data show that the dimeric complex IV is stable. The data also reveal that there are fewer protein-protein ion pairs between complex IV monomers in the presence of cardiolipins at the interface, however cardiolipin could also function as glue forming charge-charge interactions with both of the monomers. Cardiolipin-complex IV interactions seem to have more significance compared to other lipid-complex IV interactions, favoring the earlier proposals that cardiolipins are possibly involved in proton uptake. The monomeric complex IV was observed to tilt 5-10 degrees with respect to the initial position of the protein and membrane normal, while for the dimeric complex IV no tilt was observed. The difference in tilt might work as a free energy barrier in dimerization. It is also suggested that cardiolipins between the monomers could reduce the possible free energy barrier in dimerization.

Understanding of these microscopic aspects by means of molecular dynamics simulations may open up new avenues to target mitochondrial dysfunctions.

Avainsanat — Nyckelord — Keywords
respiratory chain, complex IV, Cytochrome *c* Oxidase, cardiolipin, molecular dynamics simulations

Säilytyspaikka — Förvaringsställe — Where deposited
Kumpulan campus library

Muita tietoja — Övriga uppgifter — Additional information

Contents

List of Tables

List of Figures

Supercomputers of CSC – Finnish IT center for science were used in the research performed for the thesis.

1

Introduction

All matter and energy originates from the Big Bang [1]. After the Big Bang, hydrogen and helium atoms were formed, and stars began to form. Nowadays inside stars, hydrogens fuse into helium and heavier elements, releasing energy. Eventually this energy gets radiated out of the star. On Earth, the radiation of Sun is absorbed as heat, and plants as well as other organisms use photosynthesis to produce chemical energy from radiation and carbon dioxide. This chemical energy is then consumed by living matter, which uses oxygen and glucose to produce adenosine triphosphate (ATP) through cellular respiration [2]. ATP works as the energy source for many vital functions in living matter. ATP is predominantly produced by the electron transport chain, which contains the terminal enzyme complex IV [3–10], also known as cytochrome *c* oxidase.

Complex IV is the last protein complex in the electron transport chain. Electrons are delivered to complex IV by cytochrome *c*. The function of complex IV is to transfer electrons to molecular oxygen, which is then reduced to water. Another key function of complex IV is the proton pumping, which is tightly coupled to the reduction of oxygen to water [6, 9]. Also, the activity of complex IV is coupled to cardiolipin, which is one of the abundant lipids in the inner mitochondrial membrane [11]. Overall, it is important to understand how complex IV functions because dysfunction of complex IV causes diseases such as different forms of myopathies [5].

Many aspects of complex IV behavior are well known [12]. However, some major functional questions remain unclear, such as: What is the function of the complex IV dimer? Are there artifacts in the determination of the protein structure [13, 14]? Could cardiolipins be involved in the proton uptake pathway [11]? Do they have an important role in the dimer formation and/or stability [13–15]? In this thesis, computer simulations were used in order to gain insights into these challenging questions.

Six different atomistic molecular dynamics (MD) simulations were performed. These simulations over a period of 2 to 3 µs were based on models

that consisted of dimeric (two simulations) or monomeric (four simulations) complex IV with lipids and detergents determined from the protein structure (two simulations) or not (four simulations). The additional sampling was given by initial rearranging of cardiolipins in the membrane. Based on these simulations, new cardiolipin binding sites are proposed. A possible energy barrier for complex IV dimerization is found. It is caused by 5-10 degree tilting of monomeric complex IV with respect to the initial position of the protein and membrane normal, while the dimeric complex IV does not tilt. Cardiolipin may act as a glue between monomers, which would lower the possible energy barrier of dimer formation caused by complex IV tilting. These results favor the paradigm that cardiolipin would participate in the proton uptake pathway, and that cardiolipins would have an important role in dimer formation of complex IV. In the future, the results may also help understanding the causes of myopathies better.

The structure of this thesis is as follows. Chapter 2 introduces the biological background needed to understand the results presented in Chapter 6. Chapter 3 reviews the theory of computer simulations. Chapter 4 discusses how the models used in this thesis were constructed, and Chapter 5 reviews the analysis methods used in this work. After discussing the results in Chapter 6, Chapter 7 concludes the work by presenting the key findings and their significance.

2

Biological background

This Chapter presents the necessary biological background in order to understand the research questions and the results presented in subsequent chapters. First we discuss the environment of the cell (2.1) and its components (solvent 2.1.1, membrane 2.1.2 and proteins 2.1.4). Then we will focus on complex IV (2.2).

All the information in this Chapter, if not stated otherwise, is from reference [2].

2.1 Cell environment

Organisms are divided into three domains: Archaea, Bacteria, and Eucarya [16]. Membrane is one component that differentiates eukaryotes from bacteria and archaea (together called prokaryotes). Prokaryotes are usually smaller in size and consist of a cell membrane, whereas eukaryotes (e.g., animals, plants, etc.) are larger and complex, and contain membrane-bound organelles such as mitochondria. There are many other critical components in the cells such as cytoskeleton, genetic material (DNA and RNA), and many different organelles but they are not discussed in this thesis.

2.1.1 Solvent

Membranes are surrounded by solvent from both sides. Solvent consists of mainly water, ions and many different dissolved or suspended molecules. 70-75% of the cell weight is due to water. Solvent allows different particles to diffuse in all three dimensions. Additionally water maintains the temperature inside the cell (the usual mammalian cell temperature is 310K), which is important for many biological processes [17].

Apart from providing a medium for diffusion of molecules, the solvent has another important role: dielectric screening. For example, DNA is negatively charged due to ionized phosphate groups but water screens the electrostatic

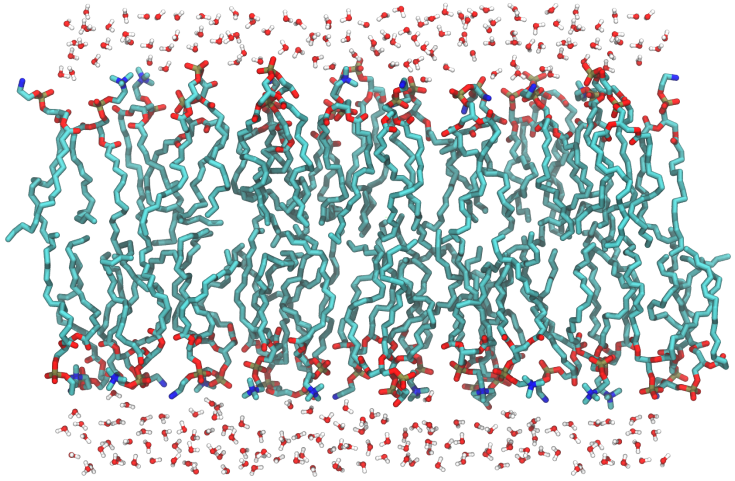


Figure 2.1: *Lipid bilayer membrane. Above and below the membrane is solvent and the headgroups interact with it. Hydrogens are hidden from the lipids for clarity.*

interactions so that the two DNA strands can twist themselves into a tight double helix [18].

A third feature of the solvent is its composition. Solvent contains proteins, salt (ions), H_3O^+ ions, and buffers. Salt or ions are critical for cellular function (e.g., Ca^{2+} in signaling), and the concentration of salt is also important for the function of many enzymes. A stable H_3O^+ concentration in water (pH) and proton gradients are extremely important for energy generation processes.

2.1.2 Membrane

Biological membranes are basically two-dimensional structures; the thickness (z-dimension) of a membrane is small (about 4 nm) and nearly constant in most circumstances, whereas the two other dimensions are much larger. One of the main functions of biological membranes is to separate one region from the other so that transport can be controlled. Biological membranes are essentially two-dimensional fluids that consist of different phospholipids forming a so-called lipid bilayer (see figure 2.1). It means that there are two layers (leaflets) of lipids. Additionally there are also small spherical membranes called liposomes and even smaller micelles. Lipids make about 3% of the total cell weight.

The phospholipids (later called just lipids) form the basic structure of all membranes. They have hydrophobic hydrocarbon chains (usually two or more) and a hydrophilic/polar headgroup (see figures 2.2 and 2.3). Because of this lipids are called as amphipathic molecules. The membrane bilayer structure

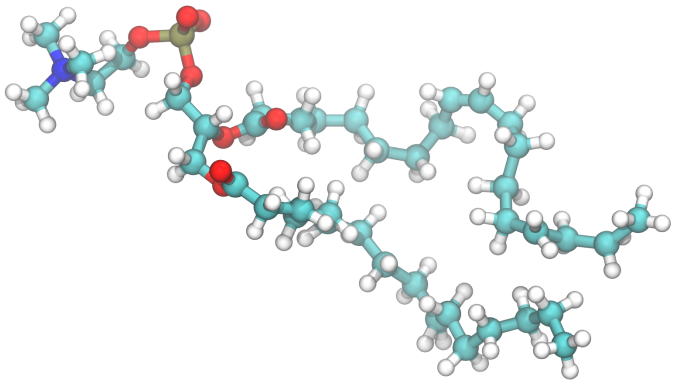


Figure 2.2: *POPC; 1-palmitoyl-2-oleoyl-sn-glycero-3-phosphocholine lipid. Cyan atoms are carbon, red denotes oxygen, mustard is for phosphorus, blue is for nitrogen, and hydrogens are white.*

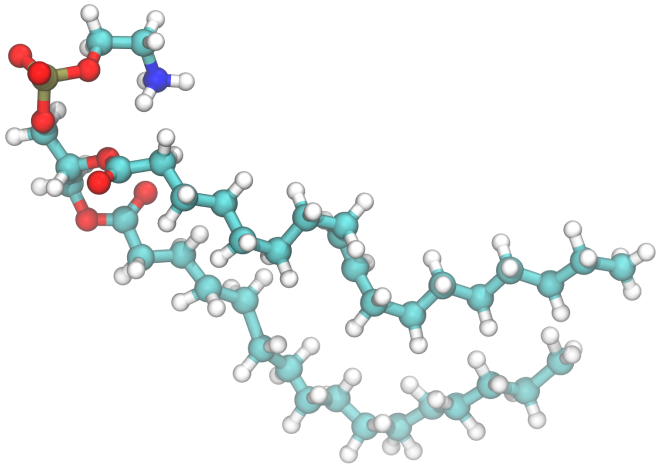


Figure 2.3: *POPE; 1-palmitoyl-2-oleoyl-sn-glycero-3-phosphoethanolamine lipid.*

Lipid type:	
phosphatidylcholine (PC)	38.4 %
phosphatidylethanolamine (PE)	24.0 %
phosphatidylinositol (PI)	16.2 %
phosphatidylserine (PS)	3.8 %
cardiolipin (CL)	16.1 %
phosphatic acid (PA) and others not detectable	1.5 %

Table 2.1: *Lipid composition of the inner mitochondrial membrane of Saccharomyces cerevisiae [19].*

is such that all the hydrophilic headgroups face the solvent on both sides and the tails of these lipids in the two leaflets interact (see figure 2.1).

Biological membranes are fluid meaning that the lipids can move past each other in terms of lateral diffusion in 2D, which may play an important role in protein dynamics and diffusion, and for establishing protein-lipid and protein-protein interactions.

2.1.3 Mitochondrial membrane

The lipid composition of membranes play an important role in the organelles. The lipid composition of the inner mitochondrial membrane is shown in table 2.1, in which therefore also highlights the main lipid components of the electron transport chain needed to generate ATP.

Cardiolipins

The inner mitochondrial membrane consists of a unique lipid-type cardiolipin, which is also found in some bacterial cell membranes [20].

Cardiolipin is a unique phospholipid with four chains [21], and sometimes it is also called diphosphatidylglycerol. Its structure is based on a reflectional symmetry assuming that the four chains are also symmetric. Both of the symmetric parts can have a negative charge so in total cardiolipin can be anionic with -2 e [22]. Cardiolipins may participate in proton transfer [20] potentially helping the cellular respiration in forming the proton-motive force (pmf) across the membrane. Additionally cardiolipins have immunological properties and are also used in the diagnosis of syphilis.

2.1.4 Proteins

Proteins can be found in membranes as well as in solvents. The function of the protein depends on the three-dimensional structure, which is the result of a unique combination and structural arrangement of 20 different amino

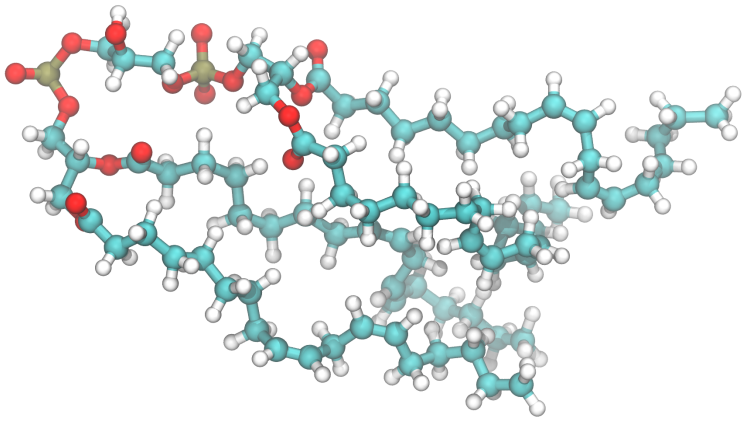


Figure 2.4: *Cardiolipin; Tetralinoleoyl cardiolipin with a charge of -2e.*

acids and their interactions during protein folding. 10-20% of the weight of a hydrated cell and about 50% of the dry weight of a cell comes from proteins.

Many proteins in cells function as enzymes that catalyze important chemical reactions, which would not occur spontaneously. Enzymes function by reducing the amount of energy (activation energy) needed to carry out the reaction. With a lower activation energy the probability of a reaction to take place increases. For example, the reduction of oxygen into water has a high activation energy, but this can be lowered with enzymes.

Other functions of proteins include, among others, transport of substances; for example, hemoglobin transports oxygen, defense against bacterial or viral infections (immunoglobulins), hormones (insulin), nutrients (casein) and structural functions (collagen and α-keratin).

Amino acids

Proteins consist of long chains of amino acids, which are connected by a peptide bond. Therefore, proteins are also called polypeptides. There are more than 200 amino acids identified in nature but only 20 different amino acids are needed for protein synthesis. These 20 amino acids are also known as common amino acids because of their frequent appearance in proteins.

All amino acids have an amino group and all the common amino acids have it attached to the α-carbon, which is right next to the carboxyl group. These are called α-amino acids, whose general formula is depicted in figure 2.5. Amino acids can be classified in many ways, and one way is to differentiate them as charged, polar, and hydrophobic, that is, based on their polarity.

A peptide bond between two amino acids is formed when an amino group

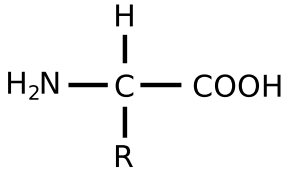


Figure 2.5: *α-amino acid general formula. R is the side chain. COOH is the carboxyl group and H_2N is the amino group. The C in the middle is the α-carbon.*

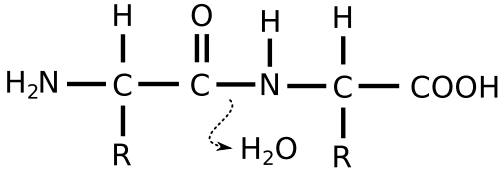


Figure 2.6: *Dipeptide. Two peptides are bonded together with a peptide bond and a water is formed. The first amino acid from left has N-terminus and the one on right has C-terminus. The charged forms are: NH_3^+ for N-terminus and COO^- for C-terminus. The same bonding logic works for all polypeptides.*

of one amino acid reacts with a carboxyl group of another amino acid, and a water molecule is formed as a side product (see figure 2.6). As long as, peptide is not cyclic (like gramicidin-S), it still has an amino (N-) terminus and a carboxy (C-) terminus, which are both usually charged in a physiological environment.

Protein folding

Polypeptide chains form, for example, α-helical (see figure 2.7) and β-sheet (see figure 2.8) domains through a process called self-assembly. The final outcome of self-assembly is highly dependent on the order of amino acids in the chain and it is driven by minimization of free energy. Sometimes there are enzymes or other proteins (chaperones) that help the protein to fold, for example, by preventing formation of aggregates and therefore speeding up the process, or by breaking disulfide linkages formed, thus preventing unwanted folding.

2.1.5 Mitochondrial proteins

Mammalian mitochondria have their own genome that encodes 13 proteins, which are all core subunits of the electron transport chain components [23].

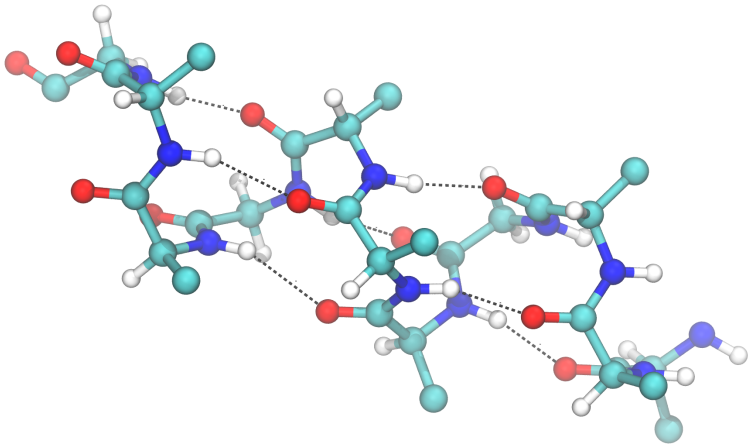


Figure 2.7: α*-helix. Only the first atom of the sidechain is shown (either a carbon or hydrogen). Dashed line denotes a hydrogen bond between the residues (i, i+4).*

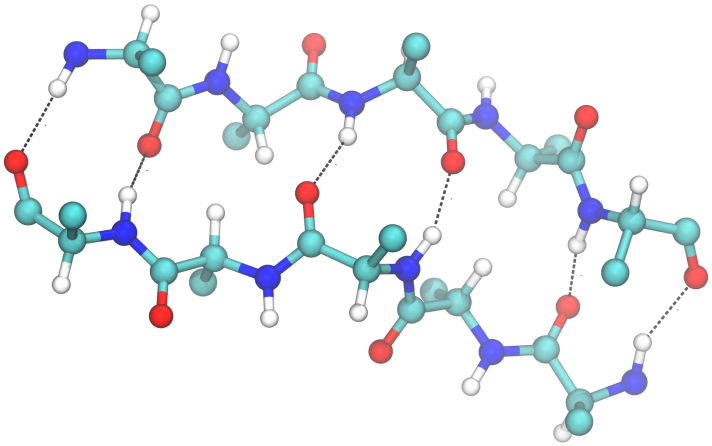


Figure 2.8: β*-sheet. Shown here is the antiparallel* β*-sheet. Antiparallel means that the direction (from the N-terminus to C-terminus) is opposite to the two peptides forming the* β*-sheet.*

Electron transport chain (ETC) includes four transmembrane proteins: complex I, II, III and IV. Additionally there is ATP synthase that produces ATP from the pmf produced by the ETC. Complex I delivers electrons to ubiquinone by oxidizing NADH, and translocates protons across the membrane. Complex II also delivers electrons to ubiquinone (Q) but does not contribute to proton pumping. Complex III receives electrons from ubiquinol (QH_2), oxidizes it, delivers electrons to cytochrome c, and produces pmf by releasing and uptaking protons from the positive and negative sides of the membrane, respectively. Finally, complex IV receives electrons from cytochrome c, oxidizes it, reduces oxygen and at the same time produces pmf, by pumping protons in a tight coupling to the oxygen reduction reaction.

2.2 Complex IV

Complex IV is the last protein complex in the electron transport chain or the respiratory chain [17]. It contains cytochrome a, cytochrome a_3, and two copper centers all of which transport electrons to an oxygen molecule. The oxygen molecule also receives four protons, and as a result two water molecules are formed (see equation 2.1).

$$8H_N^+ + 4e_P^- + O_2 \rightarrow 2H_2O + 4H_P^+, \tag{2.1}$$

where subscript N stands for the negative side of the membrane and P for the positive side. Complex IV also translocates four protons across the inner mitochondrial membrane per cycle [5].

Mitochondrial complex IV consists of at least 13 different protein subunits [24] (see also [25]). Bacterial complex IV, in contrast, consists of a smaller number of subunits (I-IV), but the three core subunits are highly conserved sequentially as well as structurally [8].

2.2.1 Structure of bovine complex IV

The structure used in this thesis is bovine heart complex IV in the fully oxidized state (PDB id 2DYR) [24]. The resolution of the structure is very accurate (1.8 Å), and it also includes crystallographically resolved lipids as well as detergent molecules. The protein is crystallized as a dimer, and each monomer contains 13 unique protein subunits. The structure also contains crystallized water molecules. Complex IV can be seen in figure 2.9. Before the determination of the structures, the ligand of metal centers as well as ion content was known [5, 10].

Subunits I, II, and III (see table 2.2) of complex IV are mitochondrially coded, and the rest of the subunits are nuclear-coded [5]. The other subunits probably stabilize the three core subunits. However, their exact function remains unknown [10]. The nomenclature of all the 13 subunits is given in table 2.2.

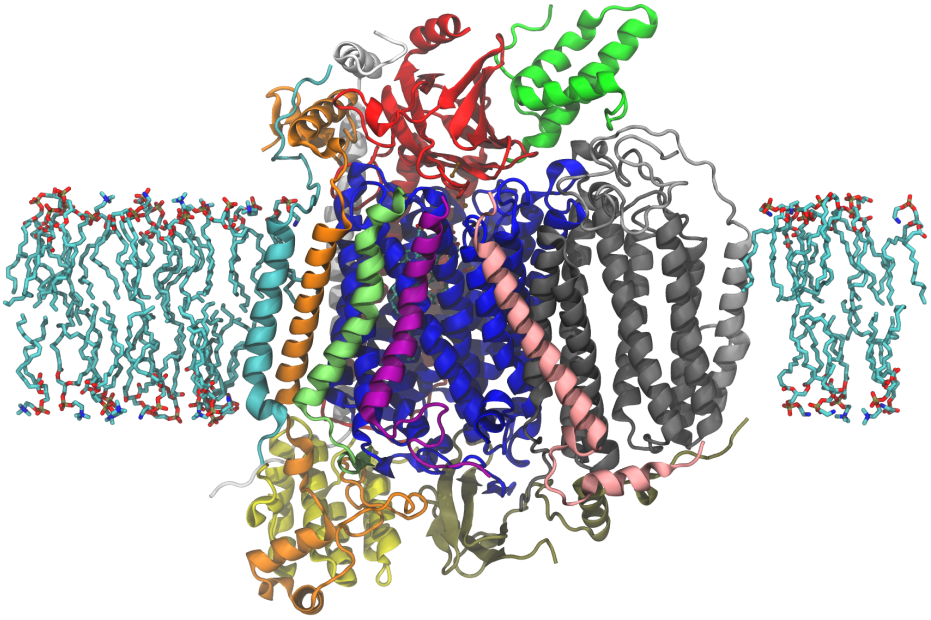


Figure 2.9: *Structure of complex IV. Each subunit has its own color. Subunit I is blue, II is red, III is gray, IV is orange, Va is yellow, Vb is tan, VIa is silver, VIb is green, VIc is white, VIIa is pink, VIIb is cyan, VIIc is purple, and VIII is lime. Part of the membrane is also shown.*

The subunits IV, Va, Vb, VIc, and VIIc (see table 2.2) in yeast are important because their deletion prevents complex IV to assemble and the activity is also diminished [5].

2.2.2 Electron transfer and proton pumping

Long-range electron transfer in the complex IV is well-known. Cu_A receives electrons from cytochrome *c*. Then the electrons are transferred to the binuclear center (BNC) via heme *a*. The BNC consists of heme a_3 and Cu_B. The binuclear center receives also oxygen and protons, following the chemical reaction where water is produced and released. Figure 2.10 shows the overall architecture of the enzyme and pathways of the proton pumping and electron transfer.

Lightly bound (or free) particles (such as valence electrons) tunnel through the medium and the rate of tunneling depends on the barrier shape, the particle's mass and on the particle's velocity [26]. Electrons tunnel about 2.5 nm through proteins at biologically relevant time scales [27]. However, protons are heavier (almost two thousand times the mass of an electron), therefore they can tunnel for shorter distances of the order of 0.6 Å [28]). In proteins, protons travel along water-wires via the Grotthuss mechanism but it could

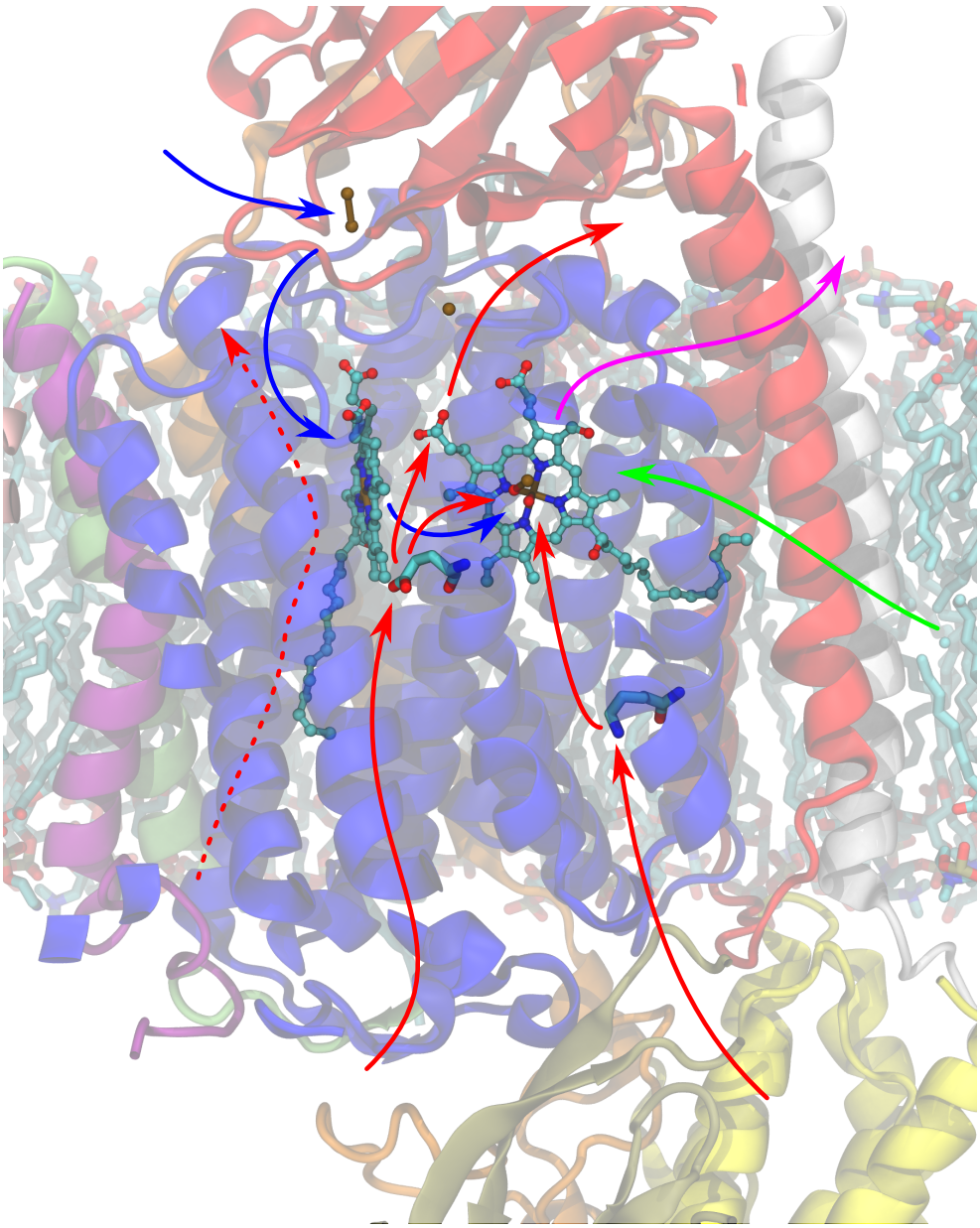


Figure 2.10: *Proton-electron transport in CcO. Blue lines are the path of electron transport. Red is for proton transport. Green for oxygen and purple for water exit. Dashed red arrow is the putative H-channel. Red arrow starting from left is the the D-channel that ends with Glu-242. From there the proton can go to PLS or BNC. The other red arrow is the K-channel that is based on Lys-319. All pumped protons use D-channel in bacterial enzymes. Electrons first go to Cu_A, then to heme a, and eventually to the BNC heme a_3 – Cu_B. Iron, copper, and magnesium are shown in brown color.*

numbers	Monomer 1	Monomer 2	Subunits	Interface
1	a	n	I	yes
2	b	o	II	yes/no
3	c	p	III	yes/no
4	d	q	IV	no
5	e	r	Va	no
6	f	s	Vb	yes/no
7	g	t	VIa	yes
8	h	u	VIb	yes
9	i	v	VIc	no
10	j	w	VIIa	no
11	k	x	VIIb	no
12	l	y	VIIc	no
13	m	z	VIII	no

Table 2.2: *Nomenclature of bovine heart complex IV. The letters of monomer 1 and 2 are from the PDB 2DYR. The interface-column tells if the subunit is part of the dimer interface. Subunit is at the interface if it is in the crystal structure within 5 Å from the other monomer. Yes/no means that not many residues are within the limit from the other monomer.*

also be possible that a protonated water cluster diffuses inside a protein for a short distance [29]. Also protons (H^+) in a biological environment are always either bound to a protein residue or, for example, as an H_3O^+ ion.

The commonly observed proton coupled electron transfer (PCET) reaction occurs differently in complex IV as compared to many other enzymes mainly because of a long-range coupling between protons and electrons, such that classical electrostatics dominate over quantum-mechanical effects. The tunneling of electrons between metal centers creates strong electric fields that polarize water molecules, which then allow proton translocation along the formed water-wire [6]. Quantum-mechanical effects are likely important for the electron and proton transfers in the active site.

Complex IV has several pathways for protons that lead to the BNC or for pumping: D, K, and H channels [10]. The D- and K-pathways are rather well-defined crystallographically as well as biochemically in both bacterial and mammalian enzymes, whereas the H channel is only functional in the mammalian enzyme albeit as a dielectric well [30]. The exit path for water molecules remains unclear although it passes through the site of the magnesium ion [31].

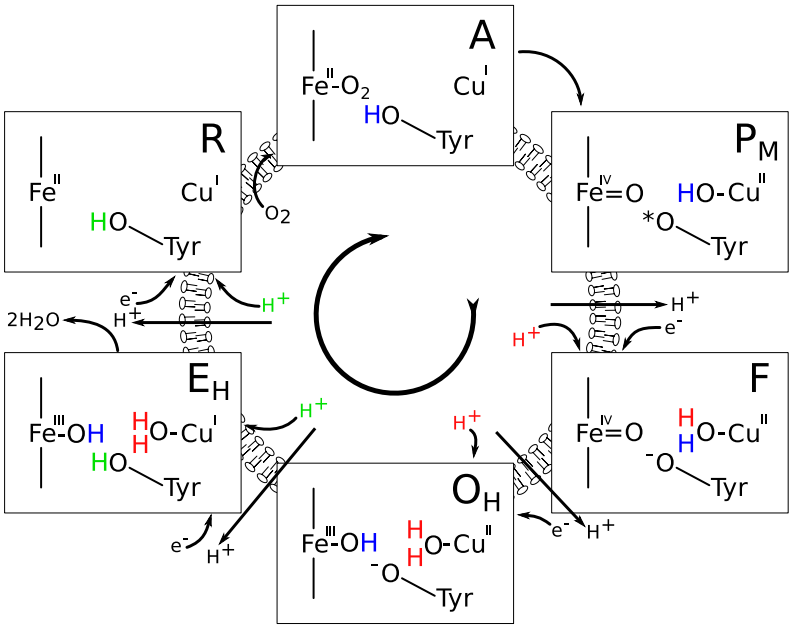


Figure 2.11: *Catalytic cycle of BNC. Protons shown in red color use the D-channel (the same as pumped protons) and the protons shown in green use the K-channel. The circle behind the boxes represents the inner mitochondrial membrane, where the protons get pumped through it. According to current understanding O_H and E_H may not have a water ligand to Cu_B [32].*

Reactions at the binuclear center

The catalytic cycle of complex IV is shown in figure 2.11. Oxygen reacts with the reduced BNC (R; see figure 2.11) forming a compound A followed by a spontaneous O-O bond scission yielding a state called P_M. The oxygen splitting reaction is followed by four PCET reactions in which a proton is consumed at the BNC and another one is pumped to the P-side [6].

2.2.3 Tightly bound cardiolipins

Wild-type complex IV contains tightly bound cardiolipins, which have been resolved crystallographically, and also analyzed using mass spectrometry. Removal of these bound cardiolipins reduces the electron transport activity approximately to half [14]. Addition of exogenous CL recovers nearly the full activity.

The removal of tightly bound cardiolipins also removes two of the protein subunits but the nearly full activity can be reached even without those subunits after adding exogenous cardiolipins to the system [14]. This suggests that cardiolipin is important for the full enzymatic activity and also for the

structural organization, but the details how this is achieved remain uncertain.

2.2.4 Complex IV as a dimer versus a monomer

Bovine mitochondrial complex IV is crystallized as a homodimer [24]. However, in supercomplex it is found as a monomer [33]. Similarly, in 2D crystals in membranes, it exists as a monomer [34]. It seems likely that the monomeric form is functional, but it is possible that it forms dimers under certain circumstances, however the details remain unclear. It has been suggested also based on simulations that cardiolipins render the formation of protein dimers possible [3, 24, 34].

3

Simulation methods

This Chapter introduces the basic idea of classical atomistic simulations (3.1), and contains information about molecular dynamics (MD) simulations in general and specific details concerning the simulation program that is used in this thesis (section 3.2).

3.1 Classical atomistic simulations

In classical atomistic simulations all the atoms are represented as spheres with properties such as charge, mass, and radii. Covalent bonds represent the permanent-like bond between bonded atoms. They are defined through relative (usually harmonic) potentials regarding the distance between atoms, the angle between bonds, and the dihedral angle between planes. For the rest of the interactions, Coulomb- and van der Waals potential are used.

Coarse-grained simulation models represent another class of classical simulation systems. They reduce complexity by treating a certain number of atoms as united beads, thereby providing a coarser picture.

3.2 MD simulations and GROMACS

In this section, the general idea of MD simulations is first discussed. Then we present a short introduction to the GROMACS software. Further, the discussion includes some basic concepts of such as molecular structure (section 3.2.1), force field parameters (section 3.2.2), interactions (section 3.2.3), and simulation techniques (section 3.2.4), followed by some of their details in the context of GROMACS. Finally the limitations of MD simulations are discussed (section 3.2.5).

One of the key ideas of MD simulations is that the particles interact with each other via non-bonded interactions such as Coulombic and van der Waals interactions as well as by bonded interactions (see below).

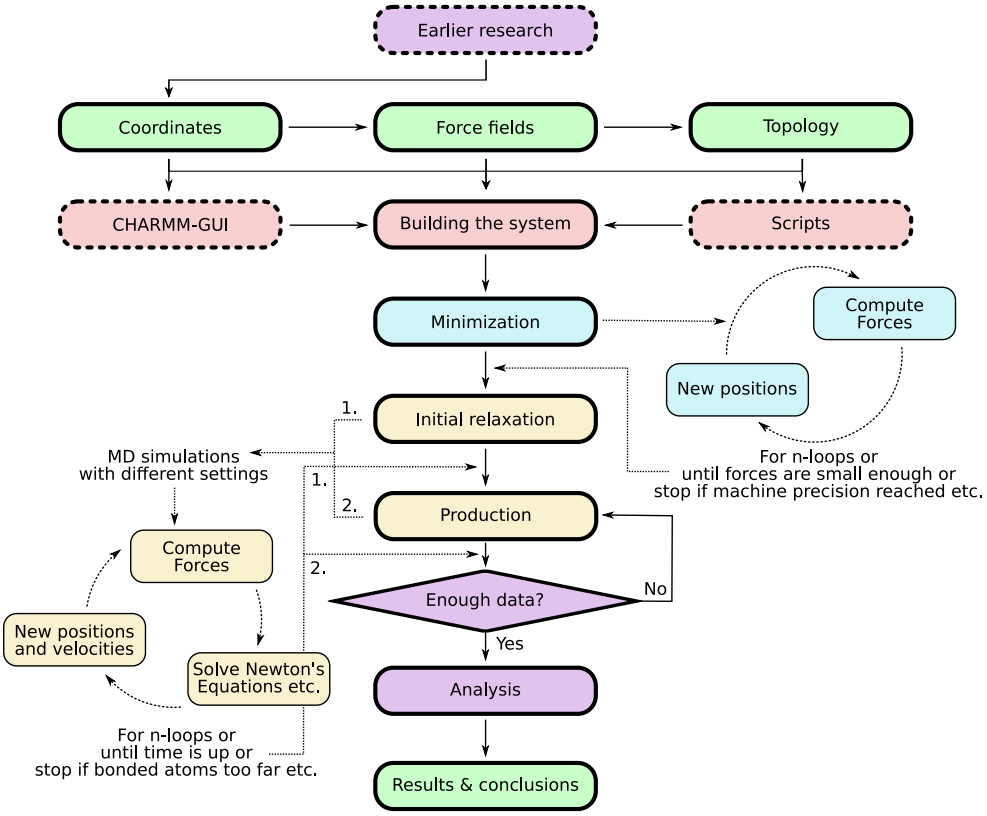


Figure 3.1: *General flow chart on using MD simulations. Coordinates, force fields, and topologies are available prior to MD simulations and they are used as a starting point for the simulations. CHARMM-GUI (see Chapter 4.1.2) is an online program and it is used in this thesis for building the system. Energy minimization removes the steric clashes and refines the positions of atoms. Initial relaxation of a solvent and a membrane are additional protocols different than the production runs.*

The interactions are determined by the coordinates of each particle (e.g., the distance between them) and by the atom types of each particle participating in the interaction. The details of interaction between each atom types are defined in the force field files (see below).

The second key aspect is the computation of forces between particles, solving Newton's equations, and updating the positions and velocities of the particles. The simulation step is repeated for a desired simulation length, and the coordinates at every n'th step are written in to a trajectory file. The simulation trajectory is analyzed by various tools (see section 5).

Flowchart in figure 3.1 describes the overall process of MD simulations.

The molecular dynamics (MD) package that is used in this thesis is the

GROningen MAchine for Chemical Simulations (GROMACS) version 5. GROMACS is open-source with free software codes and it is one of the most widely used biomolecular simulation program [35].

If not stated otherwise, the information in the rest of this Chapter is from reference [36].

3.2.1 Molecular structure

Molecular structure of a protein is the starting point for all molecular simulations. Molecular structure, as determined e.g. by x-ray crystallography, nuclear magnetic resonance (NMR), or cryo-electron microscope (CryoEM), represents a three-dimensional arrangement of all atoms in the protein. This structure is fed into the MD simulations, as described in flowchart (figure 3.1).

3.2.2 Force field parameters

A large number of parameters are needed to perform MD simulations, which are determined experimentally or by QM calculations. These are defined in force fields and topology files (in GROMACS) that are used in evaluating forces and energies between each particle of the system.

Force fields

Force fields contain all the parameters used in the bonded and non-bonded interaction calculations during a simulation (see equations 3.3-3.9). These parameters (force constants, equilibrium values of bonds, angles, etc) are different for different combinations of atoms (or atom types). Atom types are defined based on the element of the atom, and its state (what atoms it is bonded to, is it ionized, etc). Additionally, force fields include information on potential functions and their derivative forms, which are used in the simulation to calculate energies and forces.

There are different types of force fields such as GROMOS, AMBER, CHARMM, and MARTINI (a coarse-grained force field), with small to large differences in parameters and their method of development. GROMACS software can be combined with any of the above-mentioned force fields.

It is important to select the force field that fits the model system.

Topologies

Topologies provide the connection between force fields and protein coordinates. In GROMACS, the topologies include particle types, atom types, masses, charges, van der Waals parameters, and parameters for several types of interactions. They also define which atoms are bonded and where the angle and dihedral potentials are used.

3.2.3 Interactions

The interactions between particles can be divided into two: non-bonded interactions and bonded interactions.

Non-bonded interactions

There are two types of non-bonded interactions in general: electrostatic and van der Waals interactions [37]. In GROMACS, van der Waals interactions include either the Lennard-Jones potential or the Buckingham potential whereas electrostatic interactions include a Coulomb or modified Coulomb potential (although it is also possible to include user defined potentials). All of these potentials are pair-additive (total interaction is a sum of interactions of each pair of atoms; eq. 3.1) and centro-symmetric (Newton's 3rd law; eq. 3.2) in GROMACS:

$$V(\vec{r}_1, \ldots, \vec{r}_N) = \sum_{i<j} V_{ij}(\vec{r}_{ij}) \tag{3.1}$$

$$\vec{F}_i = -\sum_j \frac{dV_{ij}(r_{ij})}{dr_{ij}} \frac{\vec{r}_{ij}}{r_{ij}} = \sum_j \vec{F}_{ij} \quad \text{and} \quad \vec{F}_{ij} = -\vec{F}_{ji}, \tag{3.2}$$

where V is the vector potential of all particles, V_{ij} is the potential function between particles i and j, $\vec{r}_{ij}$ is the vector from particle i to j, $\vec{F}_i$ is the total force acting on particle i, and r_{ij} is the distance between particles i and j. Additionally, $\vec{F}_{ij}$ is the force that is caused by particle j and that acts on particle i.

The Lennard-Jones and Buckingham potentials are two types of van der Waals potential that represent a strong repulsion at short distances, and a weak attractive interaction at slightly larger distances, the key difference being in the repulsion part. The Lennard-Jones potential can be written as:

$$V_{LJ}(r_{ij}) = \frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \tag{3.3}$$

and the Buckingham potential as:

$$V_{bh}(r_{ij}) = C_{ij} \exp(-D_{ij} r_{ij}) - \frac{E_{ij}}{r_{ij}^{6}}, \tag{3.4}$$

where A_{ij}, B_{ij}, C_{ij}, and E_{ij} are constants that depend on the atom types i and j. The Buckingham potential has a more realistic repulsion term but the exponent function makes it computationally more expensive.

The Coulomb potential has the familiar form:

$$V_c(r_{ij}) = f \frac{q_i q_j}{\varepsilon_r r_{ij}}, \qquad f = \frac{1}{4\pi\varepsilon_0} = 138.935485, \tag{3.5}$$

where q_i is the (partial) charge of particle i and ε_r is the relative dielectric constant that can be defined. In GROMACS, the default value for ε_r is 1.

In order to speed up the simulations, it is recommended to use a cut-off in calculating non-bonded interactions. The cut-off means that the interactions are not calculated between particles that have a distance larger than the cut-off. However problems arise because electrostatic interactions are long ranged, and abrupt truncation may lead to errors in dynamics and energy evaluation. A much better solution is to use a shift or switch function (equations below) to modify the potentials and their derivatives to approach zero at the cut-off. Despite this smoothing, the long-range effect persists and needs to be taken into account by using algorithms such as PPPM [38] (Particle-Particle-Particle-Mesh), Ewald, or PME [39] (Particle mesh Ewald). The cutoff is also applied to van der Waals interactions but the effect is much smaller compared to electrostatics. Neighbor lists are used in order to avoid calculating interactions between particles with a distance larger than the cut-off, and update frequency of neighbor lists during simulation can be used to speed up.

The 'switch' and 'shift' functions [40] are overall similar but with some differences. A switch function multiplies the potential with a function to achieve continuation also at the cut-off, whereas the shift function adds a function to the potential. They are written as:

$$S(r) = \begin{cases} 0 & \text{if} \quad r < r_1 \\ A(r-r_1)^2 + B(r-r_1)^3 & \text{if} \quad r_1 \leq r \leq r_c \end{cases}, \tag{3.6}$$

where S is the shift function that is added to the original function and A and B are given values such that the resulting function and its derivative go to zero at the cutoff r_c. The shifting starts at r_1. Both r_c and r_1 are given as parameters when using a cut-off.

Bonded interactions

Two atoms that are bonded to each other are usually much closer than the non-bonded ones due to the formation of a bond, and due to this proximity their non-bonded interactions are excluded to avoid large repulsive interactions.

There are three types of bonded interactions: bond stretching, bond angle, and dihedral angle (see figure 3.2). All of these three categories include several different potentials that depend on either the distance of two atoms (bond stretching) or an angle between lines (bond angle) or between planes (proper/improper dihedral angle). The simplest potential that all of these categories (except proper dihedral) include is the harmonic potential:

$$V(x) = \frac{1}{2} k_x (x - x_0)^2, \tag{3.7}$$

where x is either distance or angle as described above (and in figure 3.2), k is the harmonic constant, and x_0 is the coordinate of reference point where the bond or the angle is in equilibrium.

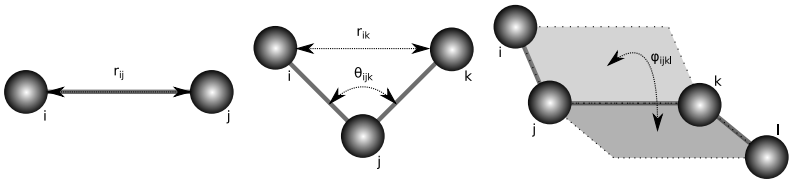


Figure 3.2: *Bonded interactions. First on the left is bond length r_{ij}. In the middle is angle θ_{ijk} and in Urey-Bradley potential also distance between i and k particles r_{ik}. On the right side proper dihedral angle ϕ_{ijkl} between planes ijk and jkl.*

In this thesis the basic harmonic potential was used for all the bonds and improper dihedrals, but for angles the Urey-Bradley potential was used:

$$V_a(\theta_{ijk}) = \frac{1}{2}k^{\theta}_{ijk}(\theta_{ijk} - \theta^0_{ijk})^2 + \frac{1}{2}k^{UB}_{ijk}(r_{ik} - r^0_{ik})^2, \tag{3.8}$$

where the atoms i and k atoms are bonded to j and the second term is a harmonic correction term on the distance between the atoms i and k. The Urey-Bradley potential was used for all angles but for some angle-types the harmonic constant k^{UB}_{ijk} is zero, therefore reducing it to the basic harmonic potential for angles.

The dihedral angle is divided into two subcategories: proper and improper dihedral angles. There are also several ways on how to select improper dihedral atoms as can be seen in figure 3.3. The purpose of improper dihedrals is to keep atoms in the same plane, or to prevent molecules from changing into their mirror images. The basic potential used for proper dihedrals is:

$$V_d(\phi_{ijkl}) = k_{\phi}(1 + \cos(n\phi - \phi_s)), \tag{3.9}$$

where ϕ is the angle between the ijk and jkl planes, ϕ_s is the reference value of the angle, k_{ϕ} the harmonic constant, and n the multiplicity term. GROMACS allows the use of multiple potentials for a single proper dihedral, and that was used in this thesis.

3.2.4 Other aspects

There are a number of further aspects that are central to MD simulations, which are discussed below.

Restraints

Restraints are a special type of potential, which can be harmonic and that is used for many different purposes such as to avoid large-scale changes in structure during the equilibration phase or to include experimental data (like from

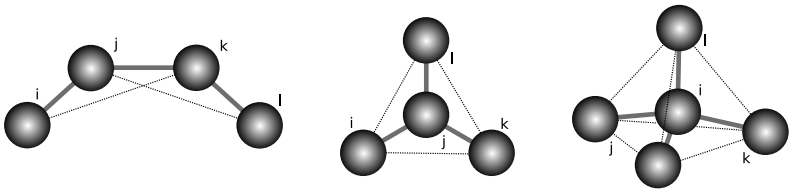


Figure 3.3: *Improper dihedrals. The first and the second case on the left illustrate an improper dihedral keeping all the atoms in the same plane. In the third example, an improper dihedral is keeping the atoms in a tetrahedral configuration.*

nuclear magnetic resonance experiments). There are four types of restraints: position, angle, dihedral, and distance restraints, and they can be applied on a single atom (position restraint) or on several atoms (other types).

Periodic boundary conditions

A biological system is not isolated, instead it is always interacting with the surrounding environment. As can be imagined, problems arise when we want to simulate a biological system (such as a protein) in atomistic detail. In a simplistic case, we can largely ignore the environment to reduce the size of the system to speed up the simulation. The downside is that a lot of biological macromolecules function in tight coupling to their surroundings. Therefore, omitting such interactions can yield large errors.

Another way is to apply some sort of approximations, such as to treat solvent as a continuum [37]. But it is well known that by replacing actual water molecules with an approximated potential results in losing some important details. Finally, the method of choice is to use periodic boundary conditions (PBCs) in which the simulation box is repeated in all three dimensions to avoid edge effects, and to describe bulk-like conditions.

There are several different PBC-box shapes that can be used in GROMACS, but they all are handled as triclinic unit cells. A triclinic unit cell is defined by three vectors $\vec{a}$, $\vec{b}$, and $\vec{c}$ such that:

$$a_y = a_z = b_z = 0 \tag{3.10}$$

$$a_x > 0, \quad b_y > 0, \quad c_z > 0 \tag{3.11}$$

$$|b_x| \leq \frac{1}{2}a_x, \quad |c_x| \leq \frac{1}{2}a_x, \quad |c_y| \leq 12b_y, \tag{3.12}$$

where the indices x, y, and z denote the components of the vectors $\vec{a}$, $\vec{b}$, and $\vec{c}$. So one of the vertices is at the origin, and the other vertices are defined by

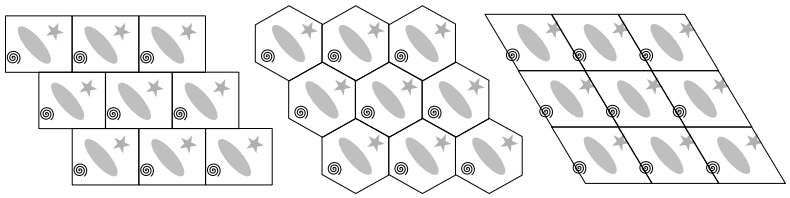


Figure 3.4: *Hexagonal periodicity. On the left is the rectangular representation. In the middle is the hexagonal (compact) arrangement. On the right side, there is the rhombus representation. In all the three representations the second box on the second row is the original box, and the eight other copies are the periodic images of the original.*

the vectors. Although the three vectors define the unit cell, different shapes can be used to visualize the same unit cell. While some shapes are better for visualizing the system (like a compact unit cell), the triclinic shape is critical for computation (because of the domain decomposition; see Parallelization) [35].

Hexagonal prism The hexagonal prism box shape has an xy-area that is 14% smaller compared to a cube with a similar image distance. This saves CPU time and also results in a more compact arrangement of the proteins. The shape is recommended for simulating membrane proteins, and it is also used in the thesis work. The box vectors of a hexagonal prism are:

$$\vec{a} = \begin{bmatrix} d \\ 0 \\ 0 \end{bmatrix}, \vec{b} = \begin{bmatrix} \pm 0.5d \\ \frac{\sqrt{3}}{2}d \\ 0 \end{bmatrix}, \vec{c} = \begin{bmatrix} 0 \\ 0 \\ z \end{bmatrix}, \quad \text{and} \begin{cases} d = p_{xy} + 2r_c \\ z = p_z + 2r_c \end{cases}, \quad (3.13)$$

where p_{xy} is the largest distance of a protein in the xy-plane (assuming protein is the largest molecule in xy-directions), r_c the cut-off radius, and p_z is the height of the protein-membrane system. The $2 \cdot r_c$ term comes from the principle that a solvent molecule should be able to interact only with one image of any molecule/complex, but this is commonly compromised in order to reduce the computational cost caused by solvents.

Three different representations for hexagonal periodicity can be seen in figure 3.4.

MD integrators

GROMACS supports several different types of MD integrators. Two most commonly used are the leap-frog algorithm [41], and two different velocity Verlet algorithms [36, 41]. In comparison to leap-frog, which is the default integrator in GROMACS, the velocity Verlet algorithm requires twice as many

communication calls making parallel computation much slower. For this reason leap-frog is used and discussed in this thesis.

The leap-frog algorithm updates positions and velocities at different times:

$$\vec{v}(t + \frac{1}{2}\Delta t) = \vec{v}(t - \frac{1}{2}\Delta t) + \frac{\Delta t}{m}\vec{F}(t) \tag{3.14}$$

$$\vec{r}(t + \Delta t) = \vec{r}(t) + \Delta t\vec{v}(t + \frac{1}{2}\Delta t), \tag{3.15}$$

where t is the simulation time, Δt is the timestep, $\vec{v}$ is the velocity vector, $\vec{r}$ is the position vector, $\vec{F}$ is the force vector of a particle, and m is the mass of the particle. The force is calculated as described in section 3.2.3, and initial positions are obtained from the coordinate file. The initial velocities can be given as parameters (for example when continuing the simulation) but they can be also randomized according to the temperature (user-defined value) by using the Maxwell-Boltzmann velocity distribution:

$$p(v_i) = \sqrt{\frac{m_i}{2\pi kT}} \exp\left(-\frac{m_i v_i^2}{2kT}\right), \tag{3.16}$$

where v_i is the x, y, or z speed component of particle i (each particle has three values for i), m_i is the mass of the particle, k is the Boltzmann's constant, T the temperature, and p denotes the probability of the specific value of speed.

Some modifications to the equations of motion described above (eq. 3.14 and 3.15) are needed when temperature and pressure coupling are applied (described below).

Temperature coupling This is needed to maintain the temperature of the system to a constant value such as 310 K, because biomolecules function at a constant physiological temperature, and therefore it is important to use temperature coupling in the simulations to mimic natural conditions. The Nosé-Hoover temperature coupling [42] was used in this study.

In the Nosé-Hoover scheme the acceleration of each particle depends not only on the forces and mass but also on the temperature of the system, as described below.

$$\frac{d^2\vec{r}_i}{dt^2} = \frac{\vec{F}_i}{m_i} - \frac{p_\xi}{Q}\frac{d\vec{r}_i}{dt}, \quad \frac{dp_\xi}{dt} = (T - T_0), \quad Q = \frac{\tau_T^2 T_0}{4\pi^2}, \tag{3.17}$$

where p_ξ is the momentum of the external heat bath, Q is the mass parameter of heat bath, T is the temperature of the system, T_0 is the target temperature, and τ_T is the period of the oscillations of kinetic energy between the system and the heat bath. The temperature and the period are the only parameters needed by the Nosé-Hoover scheme, and then the momentum of heat bath is solved from a differential equation during the simulation.

An important thing to notice is that the temperature will oscillate around T_0 for much longer time compared to some other thermostats using an exponential relaxation such as the Berendsen temperature coupling [43]. This means that Nosé-Hoover might not be the best thermostat for equilibration purposes, however it is appropriate for production runs.

Pressure coupling The biochemical reactions take place at constant pressure. Therefore, it is also important to control and reproduce the pressure in the simulations by using pressure coupling. In this study we used two different kinds of barostats: Berendsen [43] and Parrinello-Rahman [44].

The shape of the simulation box (especially size) is modified by the barostats, and the pressure can be coupled in different ways. The simplest coupling type is isotropic, in which only the box size changes. For membrane systems, semi-isotropic coupling is usually preferred because it allows the box height and xy-cross section to change independently from each other. If one would use isotropic pressure coupling with a membrane system, then there is a possibility that the box grows too much and the membrane falls apart. A more general pressure coupling type is called anisotropic and it can allow the box to change not only independently in the x-, y-, and z-directions but also the angles between box vectors are allowed to change, which may lead to an unwanted deformation of the simulation box.

In general, the Berendsen barostat is preferred for equilibration, whereas the Parrinello-Rahman method is used for production runs. This approach was also used in this thesis.

The Berendsen barostat changes the coordinates of each particle. The technique is designed so that the change in pressure is directly proportional to the difference of the actual pressure and the reference pressure:

$$\frac{d\boldsymbol{P}}{dt} = \frac{\boldsymbol{P}_0 - \boldsymbol{P}}{\tau_p}, \tag{3.18}$$

where $\boldsymbol{P}$ is the actual pressure tensor/matrix, $\boldsymbol{P}_0$ the reference pressure tensor, and τ_p is the time constant of the exponential relaxation. If pressure coupling is isotropic then the pressure tensors can be replaced by pressure scalars ($P = \mathrm{Tr}(\boldsymbol{P})/3$) and equation 3.18 then represents nine separate differential scalar equations. These equations change the shape of the box, rather than size. The velocities, however, remain unchanged.

In contrast to the Berendsen technique, the Parrinello-Rahman technique generates the correct NpT ensemble. In addition to shape, it also modifies the velocities of the particles. The box vectors change according to the following equation:

$$\frac{d^2\boldsymbol{b}}{dt^2} = V\boldsymbol{W}^{-1}\boldsymbol{b}'^{-1}(\boldsymbol{P} - \boldsymbol{P}_0), \quad (\boldsymbol{W}^{-1})_{ij} = \frac{4\pi^2\boldsymbol{\beta}_{ij}}{3\tau_p^2 L}, \tag{3.19}$$

where $\boldsymbol{b}$ is the matrix formed from box vectors, V is the volume of the box, $\boldsymbol{W}$ is a matrix parameter that determines the strength of the coupling, $\boldsymbol{\beta}$ is a matrix consisting of isothermal compressibilities, τ_p the pressure time constant, and L is the largest box matrix element. Just as with the Nosé-Hoover thermostat, the Parrinello-Rahman barostat leads to oscillations of box size (and shape), and therefore it is not used for the initial relaxation or equilibration purposes. If the pressure is initially very far from the equilibrium value, the simulation may even crash.

One similarity between Nosé-Hoover and Parrinello-Rahman is that both modify the equations of motion in a similar manner. However, there is also an important difference because the Parrinello-Rahman technique uses relative coordinates $\vec{s}_i$ for the equation of motion:

$$\frac{d^2\vec{s}_i}{dt^2} = \boldsymbol{b}'^{-1}\frac{\vec{F}_i}{m_i} - \boldsymbol{G}^{-1}\frac{d\boldsymbol{G}}{dt}\frac{d\vec{s}_i}{dt}, \quad \vec{r}_i = \boldsymbol{b}'\vec{s}_i, \quad \boldsymbol{G} = \boldsymbol{b}\boldsymbol{b}', \tag{3.20}$$

where $\boldsymbol{G}$ is the metric tensor [45].

Constraint algorithms

Bond stretching vibrations of hydrogens in real molecules are mostly in their quantum-mechanical ground state (meaning that the bond is not vibrating), and therefore classical MD simulations with potentials for these bonds are not correct [46]. A better representation for the ground state is a constraint that determines the bond length instead of a classical potential that allows the bond length to change. Another reason to use constraints is to get rid of the fastest vibrations of the bonds and angles between atoms allowing a larger time step.

In GROMACS there are two constraint algorithms: SHAKE and LINCS. In this study we have used the LINCS algorithm. LINCS allows the use of domain decomposition in parallel computing and it is also slightly faster and more stable compared to SHAKE [47]. Downside of LINCS is that it cannot be used with all the angles, as only isolated angles can be constrained.

In the LINCS algorithm, the first time step takes place without any constraints. Subsequently, LINCS resets the constrained bonds to their correct lengths. This resetting is done in two steps. First, it compares the new coordinates to the old ones, and changes the new coordinates so that the distance parallel to the old bond is the same as the length of the old bond (distance perpendicular to the old bond remains unchanged). Then, by using the angle between the new bond and the old bond, the algorithm calculates a third set of new coordinates so that the final bond length is the correct one:

$$p = \sqrt{2d^2 - l^2} \quad \text{and} \quad l = \frac{d}{\cos\theta}, \tag{3.21}$$

where p is the distance of the final coordinates parallel to the old bond, d is the constrained bond length, and θ is the angle between the changed new

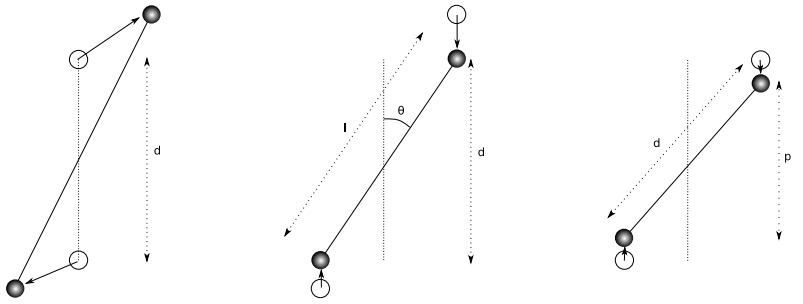


Figure 3.5: *The two updates of the LINCS constraint. On the left the hollow circles are the positions of the constrained particles in the previous step and the new unconstrained positions are shown as dark circles. In the middle, the particles are moved parallel to the old bond so that their parallel distance is d (the length of old bond) and l is the new length. The second correction sets the parallel length to p (see eq. 3.21) and the true bond length is d.*

bond and the old bond (see figure 3.5). The final bond length is the same as before the time step, only the direction is different.

Energy minimization

Before equilibration and production phases, energy minimization is usually needed in order to get rid of large forces that occur due to steric clashes, for instance. In this thesis, the steepest descent algorithm implemented in GROMACS was used for energy minimization.

The steepest descent energy minimization is not very efficient, but it is a simple and robust minimizer. This method of minimization calculates all the forces, and moves the atoms in that direction:

$$\vec{r}_{n+1} = \frac{\vec{F}_n}{\max\left|\vec{F}_n\right|} h_n, \tag{3.22}$$

where h is the maximum step originally given by the user and max (maximum) -function gives the largest absolute value of the force vector. All x, y, and z directions are handled separately. After calculating the new coordinates, the algorithm checks if the potential energy is smaller than the previous one. If yes, the new coordinates are accepted and the value of h is multiplied by a factor of 1.2 ($h_{n+1} = 1.2h_n$). Otherwise the coordinates are rejected and new ones are calculated with a smaller h ($h_n^{(\text{new})} = 0.2h_n$).

Apart from the maximum step h, the user provides a value for the number of iterations (max value for n) and an upper limit for the forces. When all the force components are smaller than the limit, the algorithm stops. If n_{max} is reached before all the forces are smaller than the limit, the iterations are

discontinued. The limit for force should not be too small in order to avoid an excessively large number of iterations, because of the inaccuracy produced by the force truncation when cutoff is used.

Infinite forces due to steric clashes could be a problem for the energy minimizer. In the steepest descent algorithm, initial infinite forces stop the iterations, unless the initial step h is large enough for the overlapping atoms to escape each other. But, if h is too large, it could also lead to infinite forces, again causing discontinuity to the process.

Parallelization

To simulate large macromolecules in a full solvent environment, it is important to use parallelization in order to reduce the wall-clock time used for a simulation. In parallelization, a task (for example, force evaluation for each particle) is divided into independent parts so that separate cores or CPUs (central processing units) can participate in computing the same task ideally, reducing the time with $1/N$, where N is the number of cores used.

In practice the wall-clock time consumed by the process is larger than the ideal case. This is because most of the processes are not directly parallelizable and extra processing is required. Another common problem is that when there is a need for communication between cores (e.g., calculating the force on atom i on core 1 due to atom j on core 2) it slows the overall process because of possible waiting times.

GROMACS uses domain decomposition in order to perform large-scale simulations in parallel. Domain decomposition means that one core handles one domain of the system and the whole system consists of several domains. The domain cores are meant for particle-particle (PP) interaction calculations, and the rest of the cores are used for long-range electrostatic (PME) calculations. Each PME core covers several PP domains. The decomposition is done at the beginning of each (selected) step so that particles that move a lot can change the PP core they were given earlier.

In order to minimize the waiting time of CPUs, GROMACS has an algorithm that does dynamic load balancing, which changes the sizes of individual domains between the time steps.

PME technique

The direct sum, to calculate electrostatics, is given by:

$$V = \frac{f}{2} \sum_{n_x} \sum_{n_y} \sum_{n_z *} \sum_{i}^{N} \sum_{j}^{N} \frac{q_i q_j}{\vec{r}_{ij,\vec{n}}}, \tag{3.23}$$

where n_x, n_y and n_z are the components of a periodic box index vector $\vec{n}$, q_i is the charge of particle i and $\vec{r}_{ij,\vec{n}}$ is the distance between particle i in the

original box and particle j in the periodic box $\vec{n}$. If all the components of $\vec{n}$ are zero (j is in the same box as i) and i equals j (particle j is the same as particle i), then we ignore the interaction between i and j (particles don't interact with themselves, only with their images and with other particles and their images) and the $*$ denotes this exclusion. The evaluation of electrostatics during a simulation via a direct sum approach is very slow.

In Ewald summation, the direct sum is replaced by two separate sums and a constant term:

$$\begin{aligned} V &= V_{dir} + V_{rec} + V_0, \\ V_{dir} &= \frac{f}{2} \sum_{i,j}^{N} \sum_{n_x} \sum_{n_y} \sum_{n_z *} q_i q_j \frac{\operatorname{erfc}\left(\beta r_{ij,\vec{n}}\right)}{r_{ij,\vec{n}}}, \\ V_{rec} &= \frac{f}{2\pi V} \sum_{i,j}^{N} q_i q_j \sum_{m_x} \sum_{m_y} \sum_{m_z *} \frac{\exp(-(\pi \vec{m}/\beta)^2 + 2\pi i \vec{m} \cdot (\vec{r}_i - \vec{r}_j))}{\vec{m}^2}, \\ V_0 &= -\frac{f\beta}{\sqrt{\pi}} \sum_{i}^{N} q_i^2, \end{aligned} \tag{3.24}$$

where β is a parameter, which determines the relative weight of the direct and reciprocal sums and $\vec{m}$ is again a periodic box index vector that has the x, y, and z components. Although this is faster than the direct summation, it is still slow (on the order of N^2 or $N^{3/2}$ depending on the technique [48]).

The actual particle-mesh Ewald (PME method) [39] does the same task as Ewald summation but it assigns the charges to a grid using interpolation. Then the grid is Fourier transformed with a 3D FFT algorithm (3-dimensional fast-Fourier-transformation) and the result in the frequency domain gives the reciprocal energy term with only a single sum over the grid. The potential is then calculated at the grid points using inverse transformation from which the forces of each atom are obtained. This is of the scale of $N \log(N)$, which is much faster than N^2 or $N^{3/2}$, and is commonly used with large model systems.

3.2.5 Limitations of MD simulations

Force field accuracy, fixed charge treatment, and the simulation lengths are some of the major limitations of the classical MD simulation approach. These introduce both systematic errors and statistical errors [49].

Methods exist to enhance sampling (replica exchange, etc) but they are still limited to models of small sizes. The polarizable force fields [50] provide a method of choice to include polarization effects, but their implementation and use remains limited at the moment. Despite these bottlenecks, classical simulations and associated free energy techniques remain methods of choice to study long time-scale behavior of biomolecules in realistic conditions, which is not yet possible with quantum-chemical or hybrid QM/MM (molecular mechanics) MD approaches.

4

Model systems

This section contains information on how the model systems of complex IV were constructed (4.1).

4.1 Setting up the simulations

A brief overview of the entire process. In this study, we used the X-ray crystal structure (PDB id 2DYR [24]) of complex IV resolved at 1.8 Å resolution containing also the crystallographically resolved water molecules, lipids, and detergents. The force field CHARMM [51] was employed for proteins, lipids, water, and ions [52,53], CHARMM-based parameters were taken from previous studies [9, 54] for metal centers, and CGenFF [55, 56] parameters for small molecules.

Some minor modifications were done in the structure files to make them compatible with softwares (4.1.1). The lipid membrane was generated with CHARMM-GUI [57] (4.1.2). After generating the membrane, PSFGEN [58] was used in order to include covalent bonds between the metal centers in complex IV and the protein ligands (4.1.3). Then the system was solvated for neutrality, and finally minimized and equilibrated (4.1.4).

4.1.1 Renaming the lipids inside the 2DYR structure

We selected the membrane composition to include POPC (1-palmitoyl-2-oleoyl-sn-glycero-3-phosphocholine), POPE (1-palmitoyl-2-oleoyl-sn-glycero-3-phosphoethanolamine), and cardiolipin (tetralinoleoyl cardiolipin with a charge of -2 e; TLCL2), and modeled the crystallographic lipids to be identical to the bulk lipids.

All PSC (2-linoleoyl-1-palmitoyl-sn-glycero-3-phosphocholine) molecules in the structure were modified to be POPC, PEK (2-arachidonoyl-1-stearoyl-sn-glycero-3-phosphoethanolamine) to be POPE, CDL (bis-(1,2-diacyl-sn-glycero-3-phospho)-1',3'-sn-glycerol; cardiolipin) to TLCL2, and PGV (2-vaccenoyl-1-palmitoyl-sn-glycero-3-phosphoglycerol) to POPG. For this, a script was

System / leaflet:	POPC	POPE	TLCL	Sums:
Dimer / upper	181	115	72	368
Dimer / lower	180	108	72	360
Dimer / total	361	223	144	728
Dimer / %	49.6 %	30.6 %	19.8 %	100.0 %
Monomer / upper	91	57	36	184
Monomer / lower	90	54	36	180
Monomer / total	181	111	72	364
Monomer / %	49.7 %	30.5 %	19.8 %	100.0 %
IMM relative %	48.9 %	30.6 %	20.5 %	100.0 %

Table 4.1: *System lipid compositions. All the dimeric model systems have the same bulk composition and so does the monomer systems. Crystallographically resolved lipids are not included here. The IMM (inner mitochondrial membrane) relative percentage is calculated from table 2.2.*

written that modified the PDB structure file so that the atom names as well as the residue names were changed as desired and extra atoms were removed. The script and other details are available in the appendix (8.1.1).

4.1.2 CHARMM-GUI

CHARMM-GUI is an online graphical user interface that prepares complex biomolecular systems for molecular simulations [57]. It was developed in 2006, and contains a module Membrane Builder (used in this thesis), which allows efficient setting up of a broad range of simulations.

The crystal structure including renamed lipids was loaded to the CHARMM-GUI Membrane Builder. Hemes, coppers, magnesium, and other ions were left out because of their complex non-standard parametrization. Also, certain additional residues in the protein were renamed such as FME (n-formylmethionine) to MET (methionine).

Terminal group patching was selected for all the protein chains and the structure was pre-oriented for the membrane.

The membrane composition, based on the inner mitochondrial lipid composition, is given in table 2.2. A sufficiently large patch of membrane-solvent was created. The final system sizes can be seen in table 4.2 (see also 4.1).

4.1.3 Building the metal centers

Metal centers were not included in the CHARMM-GUI setup, instead they were included with the structure building tool PSFGEN [58], because CHARMM-GUI had slightly minimized the system, and the important residues (ligands of metals) were off from their original crystallographic positions (pointing the

	A & B	Z	Angle 1	Angle 2	Angle 3
Dimer	209.977019	157.603	90	90	120
Monomer	149.253001	157.603	90	90	120

Table 4.2: *System dimensions and angles. A & B tells the length of the two box vectors parallel with the membrane and Z is the height of the box. The angles 1, 2, and 3 are between the box vectors (hexagonal prism). The dimensions are in ångströms and degrees.*

metal centers). For this reason, the subunits interacting with metal centers (I, II, and III) were replaced with the ones in the original crystal structure. Additionally, a few lipids were inserted between the monomers in dimeric systems to compensate for the empty region. The used patches can be seen from table 8.1 in appendix.

4.1.4 Solvation, minimization, and equilibration

Since the PSFGEN script produced files with only crystallographic water molecules but not the bulk water or ions, VMD [59] (visual molecular dynamics) was used for the solvation and ionizing the model system (see 8.1.2). The PSF (protein structure file; generated by PSFGEN) file was changed into ITP (include topology; part of GROMACS topology files) files with the psf2itp.py script (from CHARMM-GUI). Using GROMACS, the solvated and ionized system was minimized in two steps: first by freezing the protein and then by minimizing the whole system. Then equilibration was done in one go.

For equilibration and further simulation, temperature and pressure coupling groups were defined. The Berendsen pressure coupling was used for equilibration and Parrinello-Rahman in the production run. Temperature coupling was the same in equilibration and production. For the parameters, see table 8.2 in appendix.

4.1.5 Simulation models

Simulation models of this thesis consisted two dimer systems and four monomer systems, and two monomer systems have the membrane rearranged. One dimer system and one monomer system included all the crystallographic lipids and detergents. One of the rearranged systems had the protein surrounded by the cardiolipins and the other had all of them rearranged far from protein. All the simulations were at least two microseconds long. The system details can be seen in table 4.3. For the rearranging of the membrane, see 8.1.3.

	dimer	crystal lipids	rearrangement	length (ns)
1	X			3027
2	X	X		3283
3				2236
4		X		2058
5			CL near	2073
6			CL far	2405

Table 4.3: *Models and simulation lengths. The first two systems are dimeric complex IV and the rest are monomeric. Systems 2 and 4 include crystallographically resolved lipids. The last two systems have the membrane rearranged. System 5 has excess cardiolipins modeled next to the protein, and system 6 has excess POPC and POPE modeled next to the protein.*

5

Analysis methods

In order to analyze simulation trajectories both ready-made and self-made scripts and programs were used (5.1).

GROMACS prints the trajectories using a rectangular shape. Therefore, GROMACS tool 'trjconv' was used with an argument 'compact' in order to transform the rectangular box into a hexagonal form, and was visualized as such. Additionally, the protein was centered and protein subunits were blocked from jumping over the periodic boundary.

Visual Molecular Dynamics [59] was used for visualization of trajectories and rendering images used in the thesis. Tachyon ray tracing library was used for the molecular images in this thesis [60].

5.1 Analysis tools and scripts

The RMSF (root mean square fluctuation) calculator uses a predefined function 'measure rmsf' for the RMSF evaluation. It is calculated with the following formula:

$$RMSF_i = \sqrt{\frac{1}{T}\sum_{t=0}^{T-1}(x_i(t) - \overline{x}_i)^2}, \quad \Delta RMSF_i = \frac{\Delta x_i}{\sqrt{T}}, \tag{5.1}$$

where $RMSF_i$ is the RMSF of particle i, T is the number of frames, t is the time value, $x_i(t)$ is the position of particle i at time t, and $\overline{x}_i$ is the average position of particle i. Prior to calculating RMSF, it is necessary to structurally align the trajectory in order to remove the translations of the whole protein. Structural alignment is done, for example, with RMSD (root mean square deviation) Trajectory Tool in VMD (see later). Additionally, $\Delta RMSF_i$ is the error of $RMSF_i$, and Δx_i is the error of x_i (assuming that each coordinate has the same error at each frame).

Membrane thickness is calculated using a TCL script (see 8.2.1). First, it calculates the average z-coordinate of each phosphorus atom of the lipids.

Then it calculates two average values from the z-coordinates of phosphorus atoms above and below the total average z-coordinate. The membrane thickness is the difference between these two averages. It can be written as:

$$Z_{avg} = \frac{1}{N}\sum_{i}^{N} z_i, \quad T_M = \frac{1}{N_{z>Z_{avg}}}\sum_{z>Z_{avg}} z - \frac{1}{N_{z<Z_{avg}}}\sum_{z<Z_{avg}} z, \tag{5.2}$$

where Z_{avg} is the middle point of membrane, N is the number of phosphorus atoms in membrane, z_i is the z coordinate of phosphorus atom i, T_M is the thickness of membrane, and the last two terms are the averages of phosphorus atoms (in membrane) above the middle point and below it. Additionally, there is an existing tool (MEMBPLUGIN [61]) that calculates the thickness using a similar approach. MEMBPLUGIN calculates the density of phosphorus atoms in z-direction and the distance between two maximum peaks is the thickness of the membrane:

$$T_M = |z_{max1} - z_{max2}|, \tag{5.3}$$

where T_M is the thickness of the membrane and z_{maxi} is the z-position of ith highest peak. The two approaches are compared below (see section 6.2.2).

Additional scripts were written for the average distance (see section 8.2.2), and number of residues within a certain distance (see section 8.2.3).

The error of an angle is calculated using formula:

$$Err = \pm \begin{cases} 180 & \text{if } \frac{\sqrt{x^2_{1\_std}+x^2_{2\_std}}}{\sqrt{N}} > \sqrt{x^2_{1\_avg}+x^2_{2\_avg}} \\ \frac{std}{\sqrt{N}} & \text{if other} \end{cases}, \tag{5.4}$$

where x_{i_avg} is average of coordinate i on a unit circle, x_{i_dev} is the deviation of the coordinate i, std is the standard deviation calculated from the individual angles, N is the number of snapshots, and Err is the error. If the average is close enough to the origin, then the angle has no meaning and the error is the whole circle.

5.1.1 Inbuilt tools

GROMACS tools used in this thesis are the deuterium order parameter, the number of contacting atoms, and the rotation matrix.

GROMACS tool 'gmx order' calculates the deuterium order parameter. The deuterium order parameter is calculated using the formula:

$$S_{CD} = \frac{3}{2}\langle\cos^2(\theta)\rangle - \frac{1}{2}, \quad \langle\cos^2(\theta_{rand})\rangle = \frac{1}{2}\int_0^\pi \sin(\theta)\cos^2(\theta)d\theta = \frac{1}{3}, \tag{5.5}$$

where S_{CD} is the order parameter, θ is the three-dimensional angle of lipid chain at each carbon separately (compared to membrane normal) and θ_{rand}

is the disordered case where all the angles (on a sphere) have the same probability. A fully ordered case ($\theta = 0$) gives $S_{CD} = 1$, and a fully disordered system yields $S_{CD} = 0$.

Number of contacting atoms is calculated using the GROMACS tool 'gmx mindist'. Groups needed by the tool are generated with the make-index tool 'gmx make_ndx'. Number of contacts can be written as:

$$N_t = \sum_j \sum_i \delta_{r_{ij}(t)<d}, \tag{5.6}$$

where N_t is the number of contacts at time t, index i goes through particles in the first group, j particles in the other group, r_{ij} is the distance between particles i and j, d is the cutoff, and $\delta_{r_{ij}(t)<d}$ is 1, if $r_{ij} < d$ is true, and otherwise 0.

The rotation matrix tool 'gmx rotmat' was used to calculate the protein tilting relative to the initial position and the membrane normal.

VMD also provides a multitude of tools used for trajectory analysis such as RMSD trajectory [62], VolMap tool, hydrogen bond, and the distance between atoms.

The RMSD trajectory tool is based on the following formula:

$$RMSD_t = \sqrt{\frac{1}{N} \sum_i^N (x_i(t) - x_i(0))^2}, \tag{5.7}$$

which is also useful for trajectory alignment and judging the stability of the system in a simplified way.

The VolMap tool allows for calculation of density or occupancy of any component averaged over an entire trajectory and its visualization as an isosurface.

6

Results

This Chapter presents the results of the work performed in the thesis project. The summary of the major results is as follows. First, in order to make GROMACS function with the CHARMM force field parameters of metal centers, a conversion of topology (CHARMM to GROMACS) was accomplished (section 6.1). Second, detailed analyses were carried out to explore the stability of protein-membrane systems (section 6.2) and to clarify the interactions that maintain the stability (section 6.3). Third, the simulations of dimeric and monomeric forms of enzymes were compared (section 6.4) to observe that they tilt differently with respect to the membrane normal.

6.1 Conversion from CHARMM to GROMACS

The CHARMM force field and the CHARMM simulation program possess a number of tools to setup complicated lipid-protein systems, including proteins which contain metal centers covalently bonded to the protein. Handling of such metal-containing complexes is non-trivial, and GROMACS did not support setting up of such systems in a solid fashion. Therefore, as a first major task, scripts and tools were generated to prepare model systems for their use in GROMACS. One of the main reasons to do so is to exploit the key strengths of GROMACS (high speed and parallelization).

Conversion from the CHARMM parameters to the GROMACS format was achieved using a Python script called 'psf2itp.py' (generated by CHARMM-GUI). This script reads all topology and parameter files (with extensions 'prm', 'rtf', and 'str') from the given directory and the given psf file (generated by PSFGEN), and yields the required itp, which GROMACS can use. The workflow of the conversion can be seen in figure 6.1.

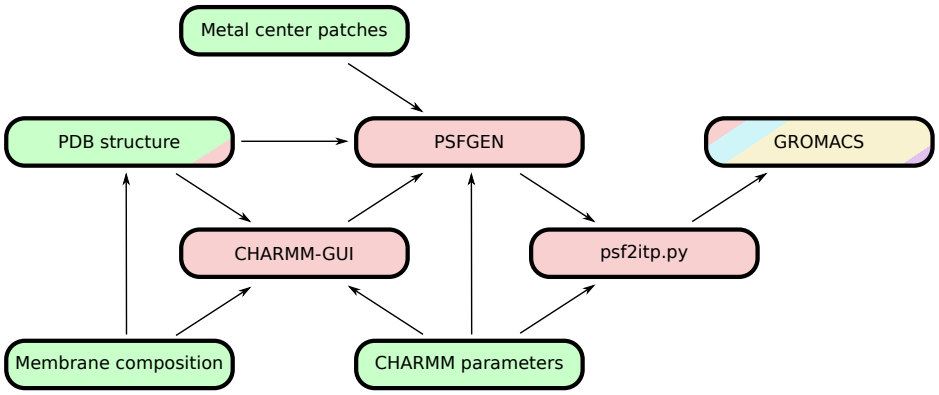


Figure 6.1: *Workflow for the conversion of the CHARMM-based setup to GROMACS. Color coding is the same as in figure 3.1. More details can be seen in section 4.1.*

6.2 Stability of the simulations and models

The stability of the model systems during a simulation can be analyzed by considering the stability of the protein (section 6.2.1), the membrane (section 6.2.2), and the entire systems (section 6.2.3).

6.2.1 Stability of the protein

In order to determine the stability of the protein, the RMSD of the monomeric (M; systems 3-6) and dimeric (D; systems 1 and 2) forms were measured (figure 6.2). Additionally, the stability of the D-form was determined by measuring the distance between the heme-irons (figure 6.3), as well as by analyzing the protein-protein interaction between monomers (figure 6.4). Also, the RMSF of separate protein subunits was measured (figure 6.5).

RMSD of the monomeric and dimeric forms of enzymes

In figure 6.2, the RMSD of each system is shown. For D-simulations, RMSD of monomers are also shown separately. All of the RMSD plots converge to a constant value, suggesting that the protein has stabilized. Some jumps do occur, but those are primarily due to the movement of a loop (for example, in system 5 at 1400-1600 ns roughly 20 residues from the end of chain J (VIIa) at the N-side of the IMM shift to a new position, and the same happens in system 3 at 1900 ns for 30 residues from the end of chain F (Vb) at the N-side).

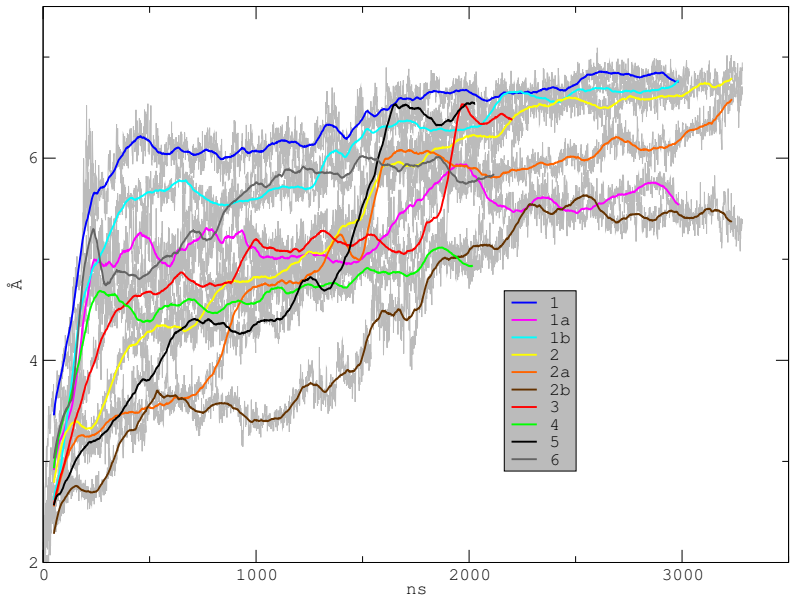


Figure 6.2: *RMSD of the monomer and dimer systems. The gray plots are the actual data and the colored lines are running averages over hundred simulation snapshots. The RMSD was calculated after aligning the protein (D and M separately). The error of RMSD in each simulation snapshot is of the order of* $\pm 10^{-4}$ *Å or less. See table 4.3 for model systems. The letters after 1 and 2 stand for the first monomer (a) and the second monomer (b) of the dimer.*

Distance of monomers in the dimeric form

Figure 6.3 shows the distance between heme *a* irons of two different monomers in a dimer simulation. The value settles to a nearly constant value, suggesting that the simulations of D-form are stable. It is also interesting to note that although the system 2 (D-form with crystallographic lipids) deviates more (roughly 5 Å) from the initial iron-iron distance (87 Å) compared to the system 1 (-3 Å), the RMSD is still smaller (see figure 6.2). This is mainly due to the space available between the two monomers in system 1 due to missing crystallographic lipids.

Protein-protein interaction between monomers

Figure 6.4 (a) shows a growing trend in the number of hydrogen bonds at the monomer-monomer interface in the D-systems. In system 1, which does not have the crystallographic lipids, the ion pairs (figure 6.4 (b)) contribute most to the interactions between the monomers in the first microsecond, while after two microseconds, the number of hydrogen bonds increases while the

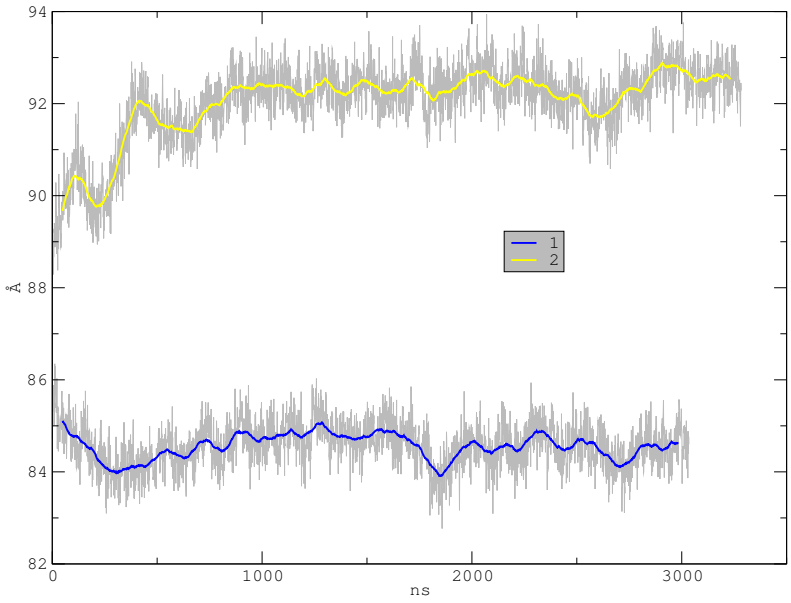
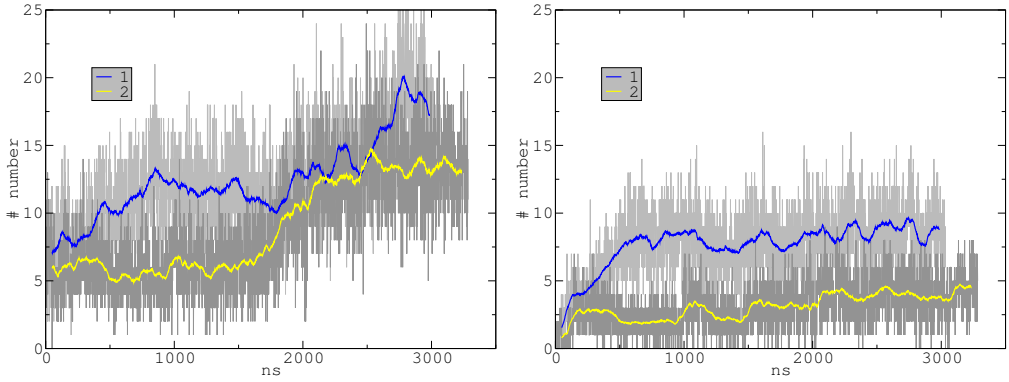


Figure 6.3: *Distance of heme a irons of two monomers in the dimeric form. Units on the y-axis are in ångströms and on the x-axis in nanoseconds. The error of each snapshot is ±0.01 Å.*



(a) *Hydrogen bonds* **(b)** *Ion pairs*

Figure 6.4: *Hydrogen bonds (a) and ion pairs (b) between the monomers in the D-simulations. Shown on the y-axis is the number of bonds as a function of time, x-axis having units of nanoseconds. The ion-pairs (b) are presented by hydrogen bonds between charged residues (backbone excluded).*

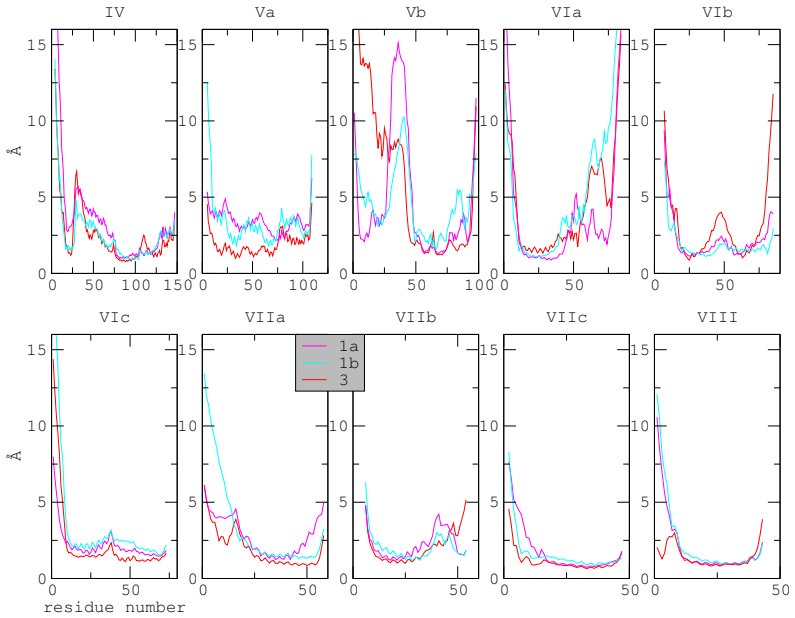


Figure 6.5: *RMSF (root mean square fluctuation) plots of all the subunits except I, II, and III from systems 1 and 3. X-axis describes the residue number and y-axis the RMSF of the C_α atom in ångströms. The error for RMSF is of the order of $\pm 10^{-4}$ Å. Protein was aligned before calculating RMSF.*

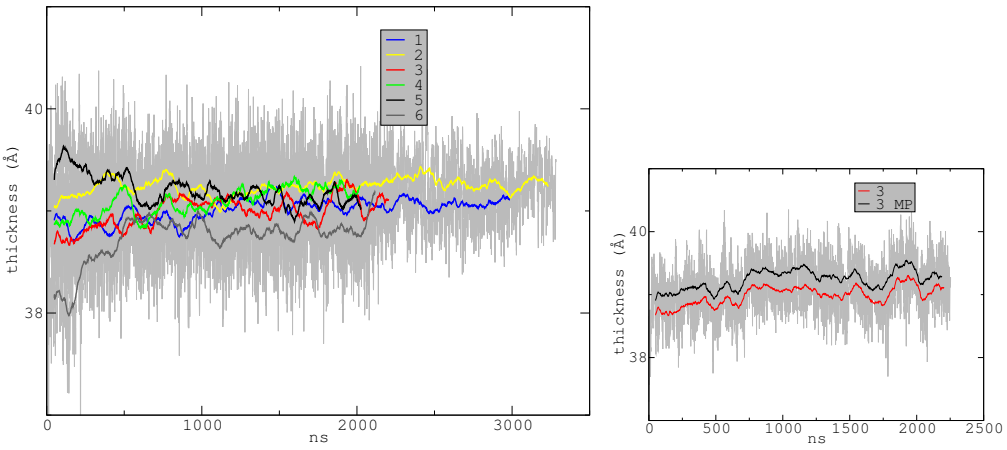
number of ion pairs remain constant. In system 2, crystallographic lipids (for example charged cardiolipins) are present, and as a result there are 3-4 ion pairs between them that remain stable. After two microseconds the total number of hydrogen bonds in system 2 increases to the same level as in system 1.

The overall data suggest that in the absence of crystallographic lipids, the interface becomes tighter due to the formation of additional interactions. On the other hand, lipids provide the monomer-monomer stability, including cardiolipin, which may stabilize the two monomers through charged lipid-protein interactions.

RMSF of different chains

Figure 6.5 shows the RMSF of different subunits in the two selected systems (1 and 3). The termini of each subunit are flexible, as expected, but the rest of the segments such as α-helices and β-sheets are stable in each subunit with only a few exceptions.

RMSF data agree with the RMSD data, because for example the system 3 had a large jump in RMSD (see figure 6.2) that was because of the first 30 residues from the subunit Vb, which has a large RMSF. One clear difference



(a) *Membrane thickness of each system*

(b) *Comparison of the average based approach and the MEMBPLUGIN tool*

Figure 6.6: *Membrane thickness measured as the average distance of phosphorus atoms in each membrane leaflet in the z-direction. Subfigure (a) shows the results from the script developed in this work, and (b) compares the result with the MEMBPLUGIN tool [61]. In subfigure (b) the MP stands for MEMBPLUGIN tool. The error of the thickness of the membrane is* $\pm 6 \cdot 10^{-4}$ *Å or less.*

between the monomeric and dimeric forms of complex IV is the subunit VIb, the "horn" at the dimer interface (at the P-side). As part of the dimer interface it cannot fluctuate freely when it is in the dimeric form.

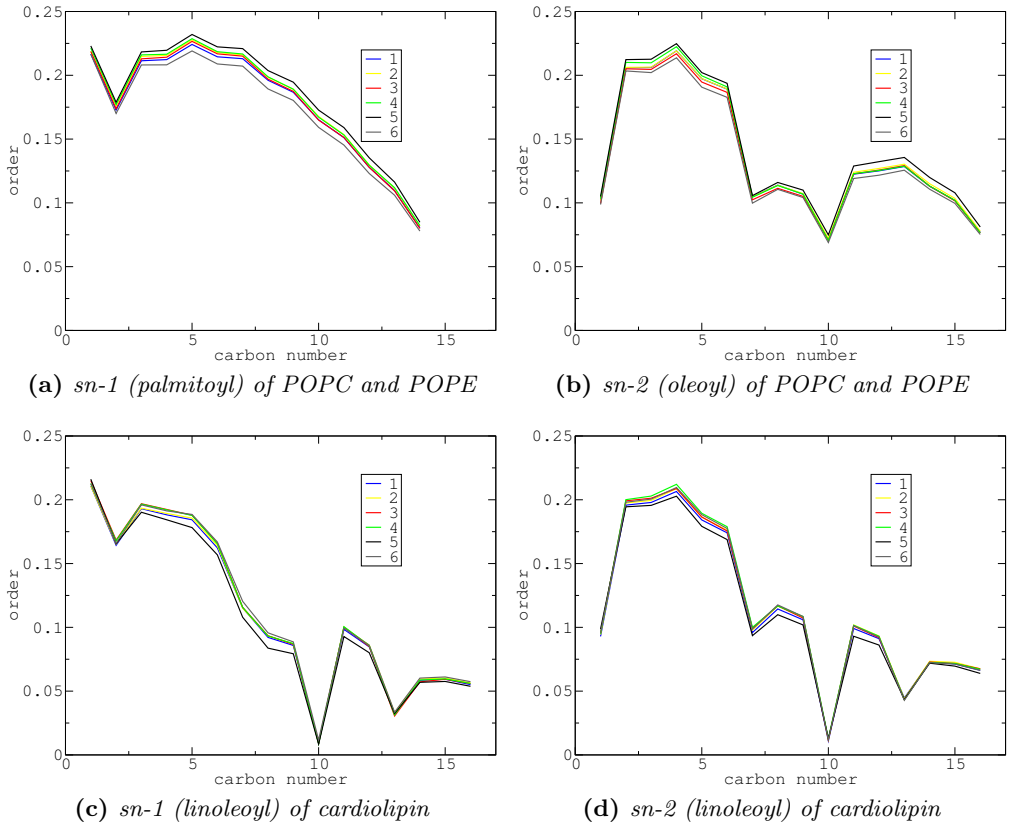
6.2.2 Stability of the membrane

The stability of the membrane was analyzed in terms of two figures of merit: membrane thickness and deuterium order parameter.

Membrane thickness

In figure 6.6 (a), the membrane thickness is plotted for all the simulation systems. In simulations with membrane lipids in a random arrangement from the beginning (systems 1-4), the thickness fluctuates around a constant (39 Å), and in systems 5 and 6, where CLs are modeled "near" and "far" from the protein, membrane thickness stabilizes to the same constant value in the first 500 ns, suggesting equilibrium to be achieved. The PC-PE-CL membrane without the protein has a thickness of 42 Å [63], and the reason why thickness in this system is smaller is most likely due to the presence of the protein.

According to Róg et al. [63] the average distance of phosphorus atoms in the opposite leaflets seems to be an inadequate definition for membrane



(a) *sn-1 (palmitoyl) of POPC and POPE*

(b) *sn-2 (oleoyl) of POPC and POPE*

(c) *sn-1 (linoleoyl) of cardiolipin*

(d) *sn-2 (linoleoyl) of cardiolipin*

Figure 6.7: *Deuterium order parameter S_{CD} of different chains. Y-axis in each plot describes the order value and x-axis describes the carbon number. Each plot is an average of two sets of chains (POPC and POPE both have identical chains, and the four chains of tetralinoleoyl-cardiolipin are identical). The error of S_{CD} is of the order of $\pm 10^{-4}$ Å.*

thickness when using a mixture of lipids. This means that both the script developed in this work and the available tool MEMBPLUGIN [61] approaches are unable to deal with hybrid lipid bilayers (for comparison, see figure 6.6 (b)).

Deuterium order parameter

The deuterium order parameter can also be used to assess the stability and flexibility of the membranes. Comparison of figures 6.7 (a) and (b) to earlier studies (see for example [64]) shows that the shape of the plots is similar. The oleoyl carbon at the position 10 has a larger value in these systems but otherwise they are very similar to earlier literature values. The linoleoyl chains

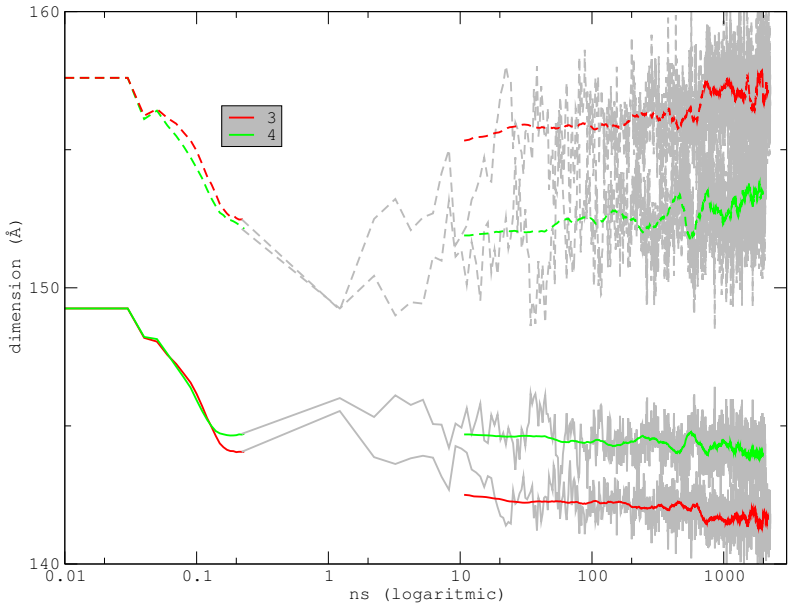


Figure 6.8: *Box dimensions from simulations of the systems 3 and 4. The dashed line is the z-dimension of the box and the continuous line is the x-dimension of the box. The initial relaxation changes to the production when the gray plots start at 0.2 ns. X-axis is in nanoseconds in a logarithmic scale because the initial relaxation is much shorter compared to the production phase. The error of each snapshot is ±0.005 Å.*

of cardiolipin in figure 6.7 (c) and (d) have smaller values at the carbons 10 and 13, as in [65], because of double bonds.

6.2.3 Fluctuation of simulation box

To further assess the stability of the simulation systems, the dimensions of simulation cells were also analyzed. Figure 6.8 shows that the Berendsen barostat (used in initial relaxation) produces exponential relaxation in the box dimensions, whereas the Parrinello-Rahman barostat (used in production simulations) fluctuates more. The box shape changes when the z-dimension grows, while the x-dimension diminishes, but the fluctuations are small, on the order of 5 Å, suggesting that the simulations of complicated membrane-protein systems are stable, despite the applied pressure.

6.3 Protein-lipid interactions

First, cardiolipin-protein interactions (6.3.1) are shown, then the interaction between all the membrane lipids and protein are compared in Chapter 6.3.2.

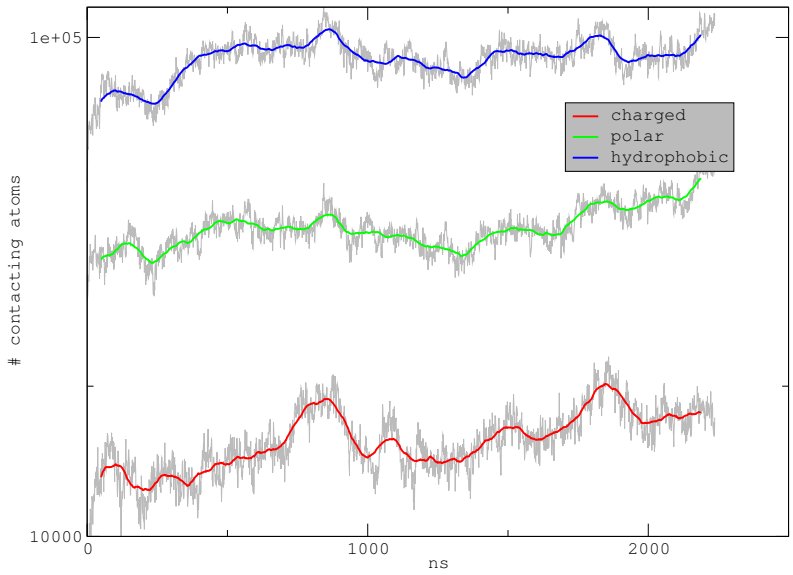


Figure 6.9: *Contacts of all CL molecules with charged, polar, and hydrophobic protein residues in the simulation of system 3. Charged protein residues are Arg, Lys, Asp, and Glu. Polar residues are Gln, Asn, His, Ser, Thr, Tyr, Cys, and Trp, whereas hydrophobic ones are Ala, Ile, Leu, Met, Phe, Val, Pro, and Gly. Number of contacting atoms (y-axis) is shown on a logarithmic scale. The x-axis is in nanoseconds. A contact is counted when one CL atom is within 8 Å of one protein atom, and each atom can have several contacts.*

6.3.1 Cardiolipin-protein interactions

First we discuss the interaction of CL with different types of protein residues, such as polar and non-polar. Second, the average distance of CL from the protein is evaluated, and finally CL occupancy on the protein surface is discussed.

Contacts between cardiolipins and the protein

Contacts between charged, polar, and hydrophobic protein residues and CL were measured for system 3. Figure 6.9 shows that most of the residues making contact with CL are hydrophobic, and charged residues have the least number of contacts with CL. This is simply due to the presence of long hydrophobic chains, which make contacts to a larger number of hydrophobic residues on the protein surface. Even though polar or hydrophobic contacts are much larger in number, it seems to be the charge-charge interactions that dominate (because shape of the plot in a logarithmic scale is comparable across orders of magnitude, assuming that the number of atoms gets separated in the constant term). This is explained by peaks just before 1000 and 2000 ns (charged

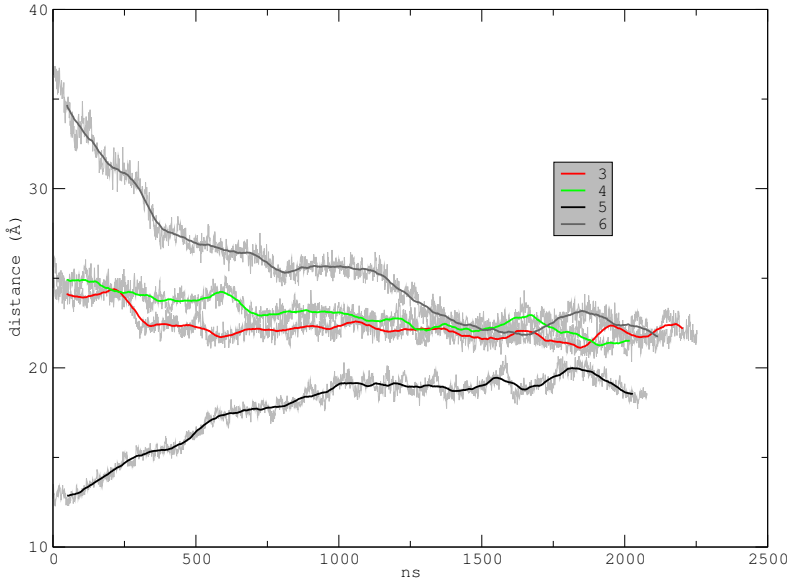


Figure 6.10: *Average distance of cardiolipins from the protein in the monomeric model systems (3-6). Periodicity is not taken into account. The error of each snapshot is* $\pm 1.2 \cdot 10^{-3}$ *Å.*

residues have the widest and tallest peaks of the three with respect to the logarithmic scale).

Average distance of cardiolipins from the protein

Figure 6.10 shows that in systems 3 and 4 the average CL-protein distance remains constant from the beginning of the simulation time. In systems 6 and 5, with a rearranged membrane, the plot levels off to a plateau at around 1 microsecond. This shows that despite modeling CL molecules "near" and "far" from the surface of the protein, the system equilibrates fast, and lipids diffuse rapidly.

Number of CLs and the protein within 3 Å of each other

Table 6.1 shows the number of CL molecules within 3 Å of the protein and vice-versa. When CL molecules are placed close to the protein, there are on average almost 10 CL molecules more close to the protein compared to the two other simulations of the monomeric form. Moreover, the data show that although the number of CL is nearly the same in system 5 as in systems 3 and 4 (34% more) the number of contacting residues is larger (85% more). Also, by comparing the D-form and M-form simulations, it is clear that the two CLs

	avg CL	avg prot.
1	43.404 ±0.003	314.688 ±0.031
2	45.138 ±0.004	395.724 ±0.012
3	28.287 ±0.004	206.504 ±0.006
4	30.442 ±0.002	222.304 ±0.017
5	39.989 ±0.003	395.920 ±0.009
6	26.149 ±0.001	140.090 ±0.042

Table 6.1: *Average number of CL and protein within 3 Å of each other. The statistics are collected from each system from 1 to 2 µs. The error values are based on the standard error of the mean.*

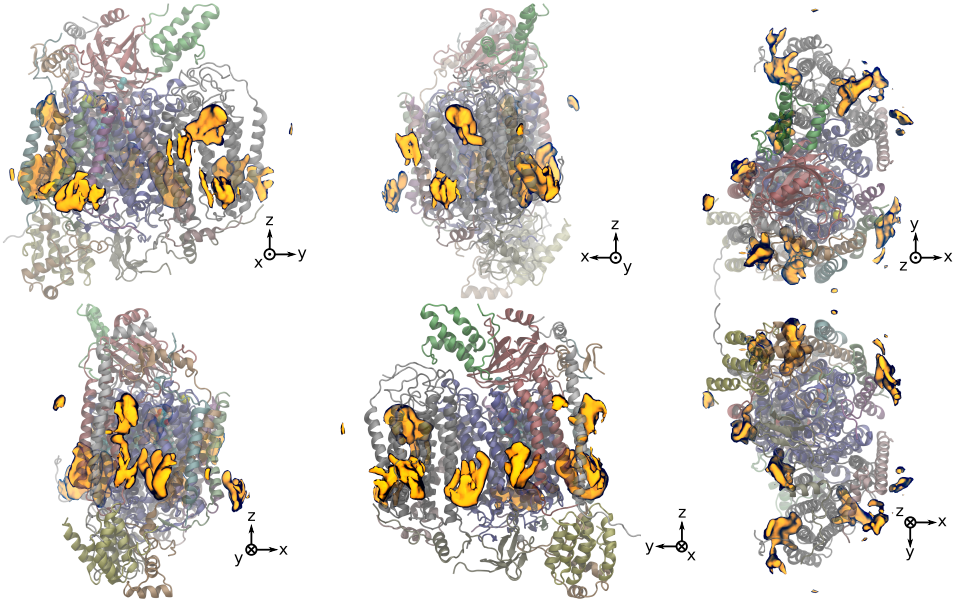


Figure 6.11: *CL occupancy of 33% calculated from the simulation system 3. Occupancy is shown with an orange surface using an isovalue of 0.33, and the protein is 25% transparent. VolMap -tool in VMD [59] was used for the occupancy of cardiolipins. The entire trajectory was aligned with the protein. Z-axis points towards the P-side of the inner mitochondrial membrane.*

located in between the monomers are in close contact with about 80 protein residues.

CL occupancy

Figure 6.11 shows that CL favors binding to some locations on protein surface. Most of these binding sites are also seen in reference [3].

	avg CL	avg POPC	avg POPE
1	57.676 ±0.001	106.180 ±0.005	67.743 ±0.003
2	59.588 ±0.002	101.669 ±0.002	62.365 ±0.013
3	39.250 ±0.002	70.153 ±0.002	45.961 ±0.004
4	41.113 ±0.000	62.427 ±0.000	42.118 ±0.006
5	45.188 ±0.000	65.256 ±0.010	37.742 ±0.005
6	36.907 ±0.007	80.487 ±0.001	41.707 ±0.004

Table 6.2: *Average number of lipids within 8 Å of the protein from 1μs to 2μs.*

6.3.2 Comparison between different lipids

We also calculated the average number of POPC, POPE, and CL molecules within 8 Å from the protein. From table 6.2 it is observed that in all the simulations there are almost the same numbers of CL and POPE molecules within 8 Å of protein. This number is also nearly the same in all monomeric simulations (average of 41.248) and dimeric simulations (average of 61.843). Average number of POPC molecules within 8 Å in systems 3 to 6 is 69.581 and in systems 1 and 2 it is 103.925. Based on these numbers and comparing these data to table 4.1, it appears that there are roughly 6 percentage points more CL near the protein than in the whole system. And, roughly 4 and 2 percentage points less POPE and POPC, respectively, near the protein compared to the total percentage of POPE and POPC.

6.4 Dimer versus monomer simulations

To understand the global dynamics of the M- and D-forms of the enzyme, tilt angle was analyzed as defined in Figure 6.12. Table 6.3 provides these data for all model systems, and shows that in monomers (systems 3 to 6) the tilt is on average 3 degrees larger compared to the monomers in dimeric complex IV. Rearranging of the membrane (systems 5 and 6) seems to produce tilt angles roughly identical to those found in the monomers 1a and 2b in dimeric simulations.

It appears that the monomeric complex IV (systems 3-6) has a certain rotation angle, whereas monomers in dimeric simulations (1a, 1b, 2a, and 2b) have different angles. This suggests that in order to form dimeric complex IV from the monomeric forms, the tilt angle has to decrease (3 degrees) and the rotation angle to change (45 degrees). This change in overall tilting (including rotation) may have an energy barrier that the monomers need to overcome.

Note that comparing the rotation angles of the b- and a-monomers, a 180 degree difference compared to the x-axis in the initial position has to be taken

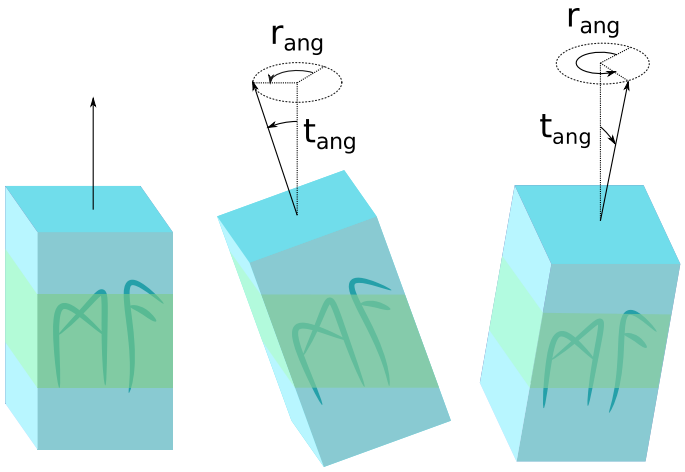


Figure 6.12: *Definitions of tilting and rotation angles. Shown on the left is the (simplified) protein in its crystallographic orientation, and the vector is parallel to the membrane normal. Comparing the middle and right orientations of the protein, the tilting angle t_{ang} is the same but the rotation angle r_{ang} has a 90 degree difference. The rotation angle is compared to the x-axis. Membrane is shown with light green color at the side of the protein.*

into account. Also, the monomeric form of complex IV has a comparable rotation angle with a-monomers of the dimeric form.

	tilt	dev	rot	dev
1	1.980 ±0.034	1.066	194.400 ±180	87.940
1a	6.734 ±0.052	1.631	146.550 ±0.437	13.825
1b	2.750 ±0.045	1.418	336.155 ±180	53.212
2	2.492 ±0.041	1.295	80.138 ±180	73.002
2a	2.534 ±0.040	1.269	271.882 ±180	66.250
2b	4.755 ±0.045	1.425	87.741 ±1.073	33.912
3	9.429 ±0.062	1.954	189.451 ±0.359	11.341
4	8.707 ±0.064	2.020	192.079 ±0.369	11.651
5	5.298 ±0.057	1.798	204.370 ±0.654	20.665
6	5.812 ±0.057	1.805	192.405 ±0.596	18.814

Table 6.3: *Tilting and rotation angles. The definitions of the tilt angle (tilt) and rotation angle (rot) can be seen from figure 6.12. Units of the angles are in degrees. Units of the deviations (dev) are in degrees, ranging from 0 to 90. The statistics are collected in each simulation from 1 to 2 μs. The error is the standard error of mean, but if the error is larger than the mean then it is 180 (see equation 5.4).*

7

Conclusions

Complex IV is the last protein complex in the electron transport chain of mitochondria and bacteria. It catalyzes oxygen reduction and couples it to proton pumping across the inner mitochondrial membrane. The pumped protons produce a proton motive force that rotates ATP synthase and allows it to produce ATP molecules, which are used to drive a number of biochemical reactions in a cell. Many key aspects of the catalytic cycle of complex IV are now known. However, understanding of the regulatory and functional aspects is still in its infancy. Understanding of these process in atomic detail would pave the way to tackle mitochondrial disorders for which limited cures exist.

To shed light on the functional aspects of this enzyme, large-scale MD simulations of monomeric and dimeric forms of complex IV were performed. The model systems were constructed from high-resolution structural data, and the full membrane-protein-solvent systems simulated in this thesis project had about 600000 atoms. To the best of our knowledge, these are the first simulations of the dimeric form of the enzyme.

Research based on these simulations aimed to answer the following questions: What is the function of the complex IV dimer assuming that complex IV dimers are formed and they are stable? Could cardiolipins be involved in the proton uptake pathway? If complex IV forms dimers, then do CLs have an important role in the dimer-formation and/or its stability?

The results suggest that the simulations are stable in microsecond time scales. This is judged by analyzing protein motions, membrane thickness, lipid order parameters, and the simulation box dimensions.

The data from the simulations show that there are 6% more CL near complex IV compared to average concentration of CL in the whole membrane. Since charge-charge interactions seem to dominate CL-complex IV interactions, these data suggest that CL may have an additional role in the function of the enzyme and not just in the putative proton uptake.

Analysis of the global motion of the protein further suggests that monomeric complex IV tilts more with respect to the crystallographic configuration and

membrane normal, and in a different direction compared to the monomers in dimeric complex IV. This means that in order to dimerize, the monomers need to change their orientation with respect to the membrane, and it also means that there could be differences in the function of the dimeric and monomeric forms of complex IV.

In the absence of CL, between complex IV monomers in dimeric complex IV simulations, protein-protein hydrogen bonds and ion pairs form in larger number, promoting stability. In the case where CL is present between the two monomers, the number of ion-pairs or hydrogen bonds decreases, but the interface is probably stabilized by polar and non-polar interactions with CL, suggesting that it acts as a glue.

The function of dimeric complex IV remains an unsolved mystery, but the investigation of the novel result of complex IV tilting could answer the question whether dimeric complex IV exists or not. This could be done if the tilting angles of complex IV would be observed experimentally, assuming that this tilting remains the same in longer time scales. The CL-glue could also help dimer formation. The tilting of monomeric complex IV introduces a possible free energy barrier between monomeric complex IV and dimeric complex IV, suggesting that there is a need for CL-glue that could bind two tilted complex IV monomers lowering the free energy barrier for dimer formation. This way CL would help in the formation of dimers.

In general, research of complex IV dimer and CL-complex IV interactions needs to be continued, and the selection of research topics is going to be done in part based on the results of this thesis. Maybe some day a breakthrough based on these results will be made relating to mitochondrial disorders.

References

[1] Arnett, D. (1996) *Supernovae and Nucleosynthesis: An Investigation of the History of Matter, from the Big Bang to the Present.* (Princeton University Press). Google-Books-ID: jeA9DwAAQBAJ.

[2] Rao, N. M. (2006) *Medical Biochemistry.* (New Age International). Google-Books-ID: EjXLThLLeCEC.

[3] Arnarez, C., Marrink, S. J., & Periole, X. (2013) Identification of Cardiolipin Binding Sites on Cytochrome *c* Oxidase at the Entrance of Proton Channels. *Scientific Reports* **3**, srep01263.

[4] Blomberg, M. R. A. & Siegbahn, P. E. M. (2015) How Cytochrome c Oxidase can Pump Four Protons per Oxygen Molecule at High Electrochemical Gradient. *Biochimica et Biophysica Acta (BBA) - Bioenergetics* **1847**, 364–376.

[5] Capaldi, R. A. (1990) Structure and Function of Cytochrome c Oxidase. *Annual Review of Biochemistry* **59**, 569–596.

[6] Kaila, V. R. I., Verkhovsky, M. I., & Wikström, M. (2010) Proton-Coupled Electron Transfer in Cytochrome Oxidase. *Chemical Reviews* **110**, 7062–7081.

[7] Ostermeier, C., Iwata, S., & Michel, H. (1996) Cytochrome c Oxidase. *Current Opinion in Structural Biology* **6**, 460–466.

[8] Schimo, S., Wittig, I., Pos, K. M., & Ludwig, B. (2017) Cytochrome c Oxidase Biogenesis and Metallochaperone Interactions: Steps in the Assembly Pathway of a Bacterial Complex. *PLoS One* **12**, e0170037.

[9] Sharma, V., Enkavi, G., Vattulainen, I., Róg, T., & Wikström, M. (2015) Proton-coupled Electron Transfer and the Role of Water Molecules in Proton Pumping by Cytochrome c Oxidase. *Proceedings of the National Academy of Sciences* **112**, 2040–2045.

[10] Yoshikawa, S. & Shimada, A. (2015) Reaction Mechanism of Cytochrome c Oxidase. *Chemical Reviews* **115**, 1936–1989.

[11] Lange, C., Nett, J. H., Trumpower, B. L., & Hunte, C. (2001) Specific Roles of Protein–Phospholipid Interactions in the Yeast Cytochrome bc1 Complex Structure. *The EMBO Journal* **20**, 6591–6600.

[12] Wikström, M., Sharma, V., Kaila, V. R. I., Hosler, J. P., & Hummer, G. (2015) New Perspectives on Proton Pumping in Cellular Respiration. *Chemical Reviews* **115**, 2196–2221.

[13] Musatov, A. & Robinson, N. C. (2002) Cholate-Induced Dimerization of Detergent- or Phospholipid-Solubilized Bovine Cytochrome c Oxidase. *Biochemistry* **41**, 4371–4376.

[14] Sedlák, E. & Robinson, N. C. (1999) Phospholipase A2 Digestion of Cardiolipin Bound to Bovine Cytochrome c Oxidase Alters Both Activity and Quaternary Structure†. *Biochemistry* **38**, 14966–14972.

[15] Sedlák, E., Panda, M., Dale, M. P., Weintraub, S. T., & Robinson, N. C. (2006) Photolabeling of Cardiolipin Binding Subunits within Bovine Heart Cytochrome c Oxidase. *Biochemistry* **45**, 746–754.

[16] Woese, C. R., Kandler, O., & Wheelis, M. L. (1990) Towards a Natural System of Organisms: Proposal for the Domains Archaea, Bacteria, and Eucarya. *Proceedings of the National Academy of Sciences* **87**, 4576–4579.

[17] Bhagavan, N. V. (2001) *Medical Biochemistry*. (Academic Press), 4 edition edition.

[18] Vasilescu, D. (2013) *Water and Ions in Biomolecular Systems: Proceedings of the 5th UNESCO International Conference*. (Birkhäuser). Google-Books-ID: Np_5BwAAQBAJ.

[19] Zinser, E., Sperka-Gottlieb, C. D., Fasch, E. V., Kohlwein, S. D., Paltauf, F., & Daum, G. (1991) Phospholipid Synthesis and Lipid Composition of Subcellular Membranes in the Unicellular Eukaryote Saccharomyces cerevisiae. *Journal of Bacteriology* **173**, 2026–2034.

[20] Hoch, F. L. (1992) Cardiolipins and Biomembrane Function. *Biochimica et Biophysica Acta (BBA) - Reviews on Biomembranes* **1113**, 71–133.

[21] Haines, T. H. & Dencher, N. A. (2002) Cardiolipin: a Proton Trap for Oxidative Phosphorylation. *FEBS Letters* **528**, 35–39.

[22] LeCocq, J. & Ballou, C. E. (1964) On the Structure of Cardiolipin. *Biochemistry* **3**, 976–980.

[23] Calvo, S. E., Clauser, K. R., & Mootha, V. K. (2016) MitoCarta2.0: an Updated Inventory of Mammalian Mitochondrial Proteins. *Nucleic Acids Research* **44**, D1251–D1257.

[24] Shinzawa-Itoh, K., Aoyama, H., Muramoto, K., Terada, H., Kurauchi, T., Tadehara, Y., Yamasaki, A., Sugimura, T., Kurono, S., Tsujimoto, K., Mizushima, T., Yamashita, E., Tsukihara, T., & Yoshikawa, S. (2007) Structures and Physiological Roles of 13 Integral Lipids of Bovine Heart Cytochrome *c* Oxidase. *The EMBO Journal* **26**, 1713–1725.

[25] Balsa, E., Marco, R., Perales-Clemente, E., Szklarczyk, R., Calvo, E., Landázuri, M. O., & Enríquez, J. A. (2012) NDUFA4 Is a Subunit of Complex IV of the Mammalian Electron Transport Chain. *Cell Metabolism* **16**, 378–386.

[26] Schleich, W. P., Greenberger, D. M., Kobe, D. H., & Scully, M. O. (2013) Schrödinger Equation Revisited. *Proceedings of the National Academy of Sciences* **110**, 5374–5379.

[27] Winkler, J. R. & Gray, H. B. (2014) Long-Range Electron Tunneling. *Journal of the American Chemical Society* **136**, 2930–2939.

[28] Masgrau, L., Roujeinikova, A., Johannissen, L. O., Hothi, P., Basran, J., Ranaghan, K. E., Mulholland, A. J., Sutcliffe, M. J., Scrutton, N. S., & Leys, D. (2006) Atomic Description of an Enzyme Reaction Dominated by Proton Tunneling. *Science* **312**, 237–241.

[29] Cukierman, S. (2006) Et tu, Grotthuss! and Other Unfinished Stories. *Biochimica et Biophysica Acta (BBA) - Bioenergetics* **1757**, 876–885.

[30] Sharma, V., Jambrina, P. G., Kaukonen, M., Rosta, E., & Rich, P. R. (2017) Insights into Functions of the H channel of Cytochrome c Oxidase from Atomistic Molecular Dynamics Simulations. *Proceedings of the National Academy of Sciences* **114**, E10339–E10348.

[31] Schmidt, B., McCracken, J., & Ferguson-Miller, S. (2003) A. discrete Water Exit Pathway in the Membrane Protein Cytochrome c Oxidase. *Proceedings of the National Academy of Sciences* **100**, 15539–15542.

[32] Sharma, V., Karlin, K. D., & Wikström, M. (2013) Computational Study of the Activated OH State in the Catalytic Mechanism of Cytochrome c Oxidase. *Proceedings of the National Academy of Sciences* **110**, 16844–16849.

[33] Althoff, T., Mills, D. J., Popot, J.-L, & Kühlbrandt, W. (2011) Arrangement of Electron Transport Chain Components in Bovine Mitochondrial Supercomplex $I_1III_2IV_1$. *The EMBO Journal* **30**, 4652–4664.

[34] Osuda, Y., Shinzawa-Itoh, K., Tani, K., Maeda, S., Yoshikawa, S., Tsukihara, T., & Gerle, C. (2016) Two-dimensional Crystallization of Monomeric Bovine Cytochrome c Oxidase with Bound Cytochrome c in Reconstituted Lipid Membranes. *Microscopy* **65**, 263–267.

[35] Abraham, M. J., Murtola, T., Schulz, R., Páll, S., Smith, J. C., Hess, B., & Lindahl, E. (2015) GROMACS: High Performance Molecular Simulations through Multi-level Parallelism from Laptops to Supercomputers. *SoftwareX* **1–2**, 19–25.

[36] Abraham, M. J., Van Der Spoel, D., Lindahl, E., & Hess, B. (2014) *The GROMACS Development Team GROMACS User Manual Version 5.0. 4*. (Accessed).

[37] Leach, A. (2001) *Molecular Modelling: Principles and Applications*. (Pearson, Harlow, England ; New York), 2 edition edition.

[38] Eastwood, J. W., Hockney, R. W., & Lawrence, D. N. (1980) P3m3dp—The Three-dimensional Periodic Particle-Particle/ Particle-mesh Program. *Computer Physics Communications* **19**, 215–261.

[39] Darden, T., York, D., & Pedersen, L. (1993) Particle mesh Ewald: An N·log(N) Method for Ewald Sums in Large Systems. *The Journal of Chemical Physics* **98**, 10089–10092.

[40] van der Spoel, D. & van Maaren, P. J. (2006) The Origin of Layer Structure Artifacts in Simulations of Liquid Water. *Journal of Chemical Theory and Computation* **2**, 1–11.

[41] Cuendet, M. A. & van Gunsteren, W. F. (2007) On the Calculation of Velocity-dependent Properties in Molecular Dynamics Simulations using the Leapfrog Integration Algorithm. *The Journal of Chemical Physics* **127**, 184102.

[42] Hoover, W. G. (1985) Canonical dynamics: Equilibrium phase-space distributions. *Physical Review A* **31**, 1695–1697.

[43] Berendsen, H. J. C., Postma, J. P. M., van Gunsteren, W. F., DiNola, A., & Haak, J. R. (1984) Molecular Dynamics with Coupling to an External Bath. *The Journal of Chemical Physics* **81**, 3684–3690.

[44] Parrinello, M. & Rahman, A. (1981) Polymorphic Transitions in Single Crystals: A new Molecular Dynamics Method. *Journal of Applied Physics* **52**, 7182–7190.

[45] Nosé, S. & Klein, M. L. (1983) Constant Pressure Molecular Dynamics for Molecular Systems. *Molecular Physics* **50**, 1055–1076.

[46] Feenstra, K. A., Hess, B., & Berendsen, H. J. C. (1999) Improving Efficiency of Large Time-scale Molecular Dynamics Simulations of Hydrogen-rich Systems. *Journal of Computational Chemistry* **20**, 786–798.

[47] Hess, B., Bekker, H., Berendsen, H. J. C., & Fraaije, J. G. E. M. (1997) LINCS: A Linear Constraint Solver for Molecular Simulations. *Journal of Computational Chemistry* **18**, 1463–1472.

[48] Toukmaji, A. Y. & Board, J. A. (1996) Ewald Summation Techniques in Perspective: a Survey. *Computer Physics Communications* **95**, 73–92.

[49] Karplus, M. & Petsko, G. (1990) *Molecular Dynamics Simulations in Biology*. Vol. 347.

[50] Halgren, T. A. & Damm, W. (2001) Polarizable Force Fields. *Current Opinion in Structural Biology* **11**, 236–242.

[51] Huang, J. & MacKerell, A. D. (2013) CHARMM36 All-atom Additive Protein Force Field: Validation Based on Comparison to NMR Data. *Journal of Computational Chemistry* **34**, 2135–2145.

[52] Jorgensen, W. L., Chandrasekhar, J., Madura, J. D., Impey, R. W., & Klein, M. L. (1983) Comparison of Simple Potential Functions for Simulating Liquid Water. *The Journal of Chemical Physics* **79**, 926–935.

[53] Beglov, D. & Roux, B. (1994) Finite Representation of an Infinite Bulk System: Solvent Boundary Potential for Computer Simulations. *The Journal of Chemical Physics* **100**, 9050–9063.

[54] Johansson, M. P., Kaila, V. R. I., & Laakkonen, L. (2008) Charge Parameterization of the Metal Centers in Cytochrome c Oxidase. *Journal of Computational Chemistry* **29**, 753–767.

[55] Vanommeslaeghe, K., Hatcher, E., Acharya, C., Kundu, S., Zhong, S., Shim, J., Darian, E., Guvench, O., Lopes, P., Vorobyov, I., & Mackerell, A. D. (2010) CHARMM General Force Field: A Force Field for Drug-like Molecules Compatible with the CHARMM All-atom Additive Biological Force Fields. *Journal of Computational Chemistry* **31**, 671–690.

[56] Zhu, X., Lopes, P. E., & MacKerell, A. D. (2012) Recent Developments and Applications of the CHARMM force fields. *Wiley interdisciplinary reviews. Computational molecular science* **2**, 167–185.

[57] Jo, S., Cheng, X., Lee, J., Kim, S., Park, S.-J, Patel, D. S., Beaven, A. H., Lee, K. I., Rui, H., Park, S., Lee, H. S., Roux, B., MacKerell, A. D., Klauda, J. B., Qi, Y., & Im, W. (2017) CHARMM-GUI 10 Years for Biomolecular Modeling and Simulation. *Journal of Computational Chemistry* **38**, 1114–1124.

[58] Gullingsrud, J., Saam, J., & Phillips, J. (2006) Psfgen User's Guide. *Urbana* **51**, 61801.

[59] Humphrey, W., Dalke, A., & Schulten, K. (1996) VMD: Visual Molecular Dynamics. *Journal of Molecular Graphics* **14**, 33–38.

[60] Ercal, F., Frank, A., Walters, G., J. Pottinger, H., & Stone, J. (2000) *An Efficient Library for Parallel Ray Tracing and Animation.*

[61] Guixà-González, R., Rodriguez-Espigares, I., Ramírez-Anguita, J. M., Carrió-Gaspar, P., Martinez-Seara, H., Giorgino, T., & Selent, J. (2014) MEMBPLUGIN: Studying Membrane Complexity in VMD. *Bioinformatics* **30**, 1478–1480.

[62] Gracia, L. (2005) *RMSDTT: RMSD Trajectory Tool.* (Weill Medical College of Cornell University, Department of Physiology and Biophysics New York).

[63] Róg, T., Martinez-Seara, H., Munck, N., Orešič, M., Karttunen, M., & Vattulainen, I. (2009) Role of Cardiolipins in the Inner Mitochondrial Membrane: Insight Gained through Atom-Scale Simulations. *The Journal of Physical Chemistry B* **113**, 3413–3422.

[64] Mendes Ferreira, T., Coreta-Gomes, F., Samuli Ollila, O. H., João Moreno, M., C. Vaz, W. L., & Topgaard, D. (2013) Cholesterol and POPC Segmental Order Parameters in Lipid Membranes: Solid State 1 H– 13 C NMR and MD Simulation Studies. *Physical Chemistry Chemical Physics* **15**, 1976–1989.

[65] Wong-ekkabut, J., Xu, Z., Triampo, W., Tang, I.-M, Peter Tieleman, D., & Monticelli, L. (2007) Effect of Lipid Peroxidation on the Properties of Lipid Bilayers: A Molecular Dynamics Study. *Biophysical Journal* **93**, 4225–4236.

8

Appendix

8.1 Preparing the systems

8.1.1 script_replace_lipidnames.sh

```
#!/bin/bash

index=0

# here the conversion_list.txt is parsed
while read line2
do
  old_name[$index]=$(echo $line2 | cut -d';' -f2)
  new_name[$index]=$(echo $line2 | cut -d';' -f3)

  if [[ ${old_name[$index]} == *"PSC"* ]]
  then
    start_psc=$index
  elif [[ ${old_name[$index]} == *"PEK"* ]]; then
    start_pek=$index
  elif [[ ${old_name[$index]} == *"PGV"* ]]; then
    start_pgv=$index
  elif [[ ${old_name[$index]} == *"CDL"* ]]; then
    start_cdl=$index
  fi

  index=$((index+1))
done<conversion_list.txt

echo $start_pek

while read line
do
  # only HETATM are changed (all lipids are HETATM)
  if [[ $line != *"HETATM"* ]]
  then
    echo "$line"
    continue
```

```
  fi

  # here it's determined that which part of conversion_list.txt is used
  if [[ $line == *"PSC"* ]]
  then
    start=$start_psc
  elif [[ $line == *"PEK"* ]]; then
    start=$start_pek
  elif [[ $line == *"PGV"* ]]; then
    start=$start_pgv
  elif [[ $line == *"CDL"* ]]; then
    start=$start_cdl
  else
    echo "$line"
    continue
  fi

  end=${#old_name[*]}
  compare=${old_name[$start]} # compare = residue name

  # the conversions are looped
  for (( i=$start; i<$end; i++ ))
  do
    old=${old_name[$i]}
    new=${new_name[$i]}

    if [[ $line == *"$old"* ]]
    then

      if [[ $new == "" ]]
      then
        #echo "found empty with £old"
        continue 2
      fi

      line=${line/$old/$new}

      # each line contains the residue name and atomname,
      # and residue name is changed first
      # then after the second hit we can stop the loop
      if [[ $old != $compare ]]
      then
        break
      fi
    fi
  done

  echo "$line"
done<$1
```

POPC from conversion_list.txt

```
;PSC ;POPC;         #here we have POPC
;N ;N ; ;#N
;P ;P ; ;#P
;O12 ;O12 ; ;#O
;O13 ;O13 ;
;O14 ;O14 ;
;O11 ;O11 ;
;O01 ;O21 ;
;O02 ;O22 ;
;O03 ;O31 ;
;O04 ;O32 ;
;C06 ;C13 ; ;#CN
;C07 ;C14 ;
;C08 ;C15 ;
;C05 ;C12 ; ;#CPN
;C04 ;C11 ;
;C03 ; C1 ; ;#COO
;C02 ; C2 ;
;C01 ; C3 ;
;C19 ;C31 ; ;#C_chain2
;C20 ;C32 ;
;C21 ;C33 ;
;C22 ;C34 ;
;C23 ;C35 ;
;C24 ;C36 ;
;C25 ;C37 ;
;C26 ;C38 ;
;C27 ;C39 ;
; C28 ;C310 ;
; C29 ;C311 ;
; C30 ;C312 ;
; C31 ;C313 ;
; C32 ;C314 ;
; C33 ;C315 ;
; C34 ;C316 ;
;C1 ;C21; ;#C_chain1
;C2 ;C22;
;C3 ;C23;
;C4 ;C24;
;C5 ;C25;
;C6 ;C26;
;C7 ;C27;
;C8 ;C28;
;C9 ;C29;
; C10 ;C210 ;
; C11 ;C211 ;
; C12 ;C212 ;
; C13 ;C213 ;
; C14 ;C214 ;
; C15 ;C215 ;
; C16 ;C216 ;
; C17 ;C217 ;
```

```
; C18 ;C218 ;
; ;***!!!--- ERROR ---!!!***; #if not found
```

...

8.1.2 script_solvate_ionize.tcl

```
#vmd -dispdev text -e script_solvate_ionize.tcl -args -filestart jou
↪  -filestart jee

set filestart "final"
set water_file "waterbox"
set hex_file "hexagonal_waterbox"
set ion_conc "0.15"
set ion_file "final_solv_ions"

set pi [expr {acos(-1)}]

set angle "$pi/6"

set x "209.977019"
set x_shift "0.0"
set y [expr 2*$x*tan($angle)]
set y_shift "-15.0"
set z "157.603"
set z_shift "0.0"

set index "0"
puts ""
while { $index<$argc-1} {
  switch -glob [lindex $argv $index] {
    "_*" {
      set varname [scan [lindex $argv $index] "_%s"]
      set varvalue [set $varname]
      if { $varvalue == "" } {
        error "$varname was empty, stopping the script"
      }
      puts "found: $varname: $varvalue"
      set index [expr $index+1]
      puts "replacing: $varvalue with [lindex $argv $index]"
      set $varname [lindex $argv $index]
    }
    default {
      puts "warning: use the script like this:
vmd -dispdev text -e script_solvate_ionize.tcl -args _filestart final
↪  _water_file waterbox
the script then sets the value of variable filestart to final and water_file
↪  to waterbox"
    }
    }
  puts ""
  set index [expr $index+1]
}
```

```
set xmax [expr $x+$x_shift]
set ymax [expr $y+$y_shift]
set zmax [expr $z+$z_shift]

set a "{{$x_shift $y_shift $z_shift} {$xmax $ymax $zmax}}"

package require solvate
eval "solvate $filestart.psf $filestart.pdb -o $water_file -minmax $a"

mol load psf $water_file.psf pdb $water_file.pdb

set function "(-x+$x/2)*tan($angle)"
set water "segname WT1 to WT9"

set radius "7"
set remove_these_waters "same residue as $water and ((y < $function+$y_shift)
↪  or (y < -$function+$y_shift) or (y > -$function+$y_shift+$y) or (y >
↪  $function+$y_shift+$y))"
set water_inside "same residue as $water and (all not within ${radius} of (all
↪  not within ${radius} of not $water))"

set remain [atomselect top "not ($remove_these_waters) and not
↪  ($water_inside)"]
set move_vector "{[expr -$x/4] $y_shift 0}"
eval "$remain moveby $move_vector"
$remain writepdb $hex_file.pdb
$remain writepsf $hex_file.psf

package require autoionize

autoionize -psf $hex_file.psf -pdb $hex_file.pdb -sc $ion_conc -o $ion_file

#without these the psf2itp will produce an error
set first_line "PSF EXT CMAP CHEQ XPLOR"
exec sed -i_backup.psf "1s/.*/$first_line/" $ion_file.psf

exit
```

8.1.3 rearrange_residues.tcl

```
puts "usage:
switch_group all_of_these with_these"

proc switch_residues_center {atom1 atom2} {
  set res1 [atomselect top "same residue as $atom1"]
  set res2 [atomselect top "same residue as $atom2"]

  set coord1 [geom_center $res1]
  set coord2 [geom_center $res2]

  set diff1 [vecsub $coord2 $coord1]
  set diff2 [vecsub $coord1 $coord2]
```

```
  $res1 moveby $diff1
  $res2 moveby $diff2

  $res2 delete
  $res1 delete
}

proc geom_center {selection} {
  # set the geometrical center to 0
  set gc [veczero]
  foreach coord [$selection get {x y z}] {
    # sum up the coordinates
    set gc [vecadd $gc $coord]
  }
  # and scale by the inverse of the number of atoms
  return [vecscale [expr 1.0 /[$selection num]] $gc]
}

proc switch_group {all_of_these with_these} {
  set sel1 [atomselect top $all_of_these]
  set sel2 [atomselect top $with_these]

  set idx1 [$sel1 list]
  set idx2 [$sel2 list]

  if { [llength $idx2] < [llength $idx1] } {
    puts "group 2 must be larger or equal with group 1"
    return
  }

  set max [llength $idx1]
  for {set i 0} {$i < $max} {incr i} {
    switch_residues_center "index [lindex $idx1 $i]" "index [lindex $idx2 $i]"
  }

  $sel1 delete
  $sel2 delete
}

# ... file contains also functions for scaling the residues, used in removing
↪  the overlapping of them
```

protonation patch	GLUP	ASPP	LSN
subunit & resid	1&242; 3&90	1&364; 3&246	1&319
dimer patch	NTN2	CNEU	-
subunit & resid	7&1	7&84	-
metal center patch	A_3	A3OH	PHEM
metal center patch	OWYM	FHEM	CUAO

Table 8.1: *Patches for protonation [51], dimer, and metal centers [9, 54]. The NTN2 patch was modified from NNEU patch [51] by changing deleted atom from HN to HT3, and by removing the duplicate bonds between atoms N, HT1, and HT2. Subunits can be seen from table 2.2. Resid means residue index. Selected metal center patches were used wherever possible. Terminal patching was done before other patching for all the subunits. GLUP means protonated glutamic acid, ASPP means protonated aspartic acid, LSN means neutral lysine, NNEU and CNEU are patches for neutral N- and C-terminus, and the metal center patches are all for fully oxidized complex IV. Dimer patches were used only with dimeric complex IV.*

parameter	value	parameter	value
integrator	md	dt	0.002
cutoff-scheme	verlet	nstlist	20
rlist	1.2	coulombtype	pme
rcoulomb	1.2	vdwtype	Cut-off
vdw-modifier	Force-switch	rvdw_switch	1.0
rvdw	1.2	tcoupl	Nose-Hoover
tc_grps	p, m & s	tau_t	1.0
ref_t	310	pcoupl	Parrinello-Rahman
tau_p	5.0	compressibility	4.5e-5
ref_p	1.0	constraints	h-bonds
constraint_algorithm	LINCS	continuation	no
nstcomm	100	comm_mode	linear
comm_grps	p, m & s		

Table 8.2: *Parameters for production run in GROMACS [36]. The only differences compared to initial relaxation are pressure coupling (Berendsen), and the generation of velocities at the temperature of 310 kelvin. The 'p, m & s' are groups for protein, membrane & solvent.*

8.2 Analysis scripts

8.2.1 membrane_thickness.tcl

```
proc avg {list_values} {
  set length [llength $list_values]
  set sum 0
  for {set i 0} {$i < $length} {incr i} {
    set sum [expr $sum + [lindex $list_values $i]]
  }
  return [expr $sum * 1.0 / $length]
}

proc avg_dist_in_z {atoms1 atoms2 frame} {
  set a1 [atomselect top "$atoms1" frame $frame]
  set a2 [atomselect top "$atoms2" frame $frame]

  set z1 [$a1 get z]
  set z2 [$a2 get z]

  set avg1 [avg $z1]
  set avg2 [avg $z2]

  set value [expr $avg1 - $avg2]

  $a1 delete
  $a2 delete

  return $value
}

proc plot_memb_thickness {filename} {
  set nf [molinfo top get numframes]
  set chan [open $filename w]
  set selection "resname POPC POPE TLCL and name P P1 P2"

  for {set i 0} {$i < $nf} {incr i} {
    set sel [atomselect top "$selection"]
    set z_sel [$sel get z]
    set middle [avg $z_sel]
    set dist [avg_dist_in_z "$selection and z > $middle" "$selection and z <
↪   $middle" $i]
    puts $chan "$i $dist"
    $sel delete
  }

  close $chan
}
```

8.2.2 nearest_distance.tcl

```
puts "example:
plot_average_distance C2_avg_dist.txt \"resname TLCL and name C2\" \"protein
↪   and name CA\""
```

```
proc calculate_nearest_distance {atoms1 atoms2 frame} {
  set sel1 [atomselect top $atoms1 frame $frame]
  set sel2 [atomselect top $atoms2 frame $frame]

  set idx1 [$sel1 list]
  set idx2 [$sel2 list]

  set max_i [llength $idx1]
  set max_j [llength $idx2]

  set min_dist 100000

  if {$max_j == 0} {
    return $min_dist
  }

  for {set i 0} {$i < $max_i} {incr i} {
    for {set j 0} {$j < $max_j} {incr j} {
      set a [lindex $idx1 $i]
      set b [lindex $idx2 $j]

      set dist [measure bond [list $a $b] frame $frame]
      if {$dist < $min_dist} {
        set min_dist $dist
      }
    }
  }


  $sel1 delete
  $sel2 delete

  return $min_dist
}

proc fast_nearest_distance {atoms1 atoms2 frame} {
  set min_dist 100000
  for {set i 1} {$i < 100000} {incr i} {
    set dist [calculate_nearest_distance $atoms1 "((${atoms2}) and within $i
↪   of (${atoms1}))" $frame]

    if {$dist < $min_dist} {
      return $dist
    }
  }
  return $min_dist
}

proc average_distance {atoms1 atoms2 frame} {
  set sel1 [atomselect top $atoms1 frame $frame]

  set idx1 [$sel1 list]
```

```
  set max_i [llength $idx1]

  set sum 0

  for {set i 0} {$i < $max_i} {incr i} {
    set sum [expr $sum + [fast_nearest_distance "index [lindex $idx1 $i]"
↪   $atoms2 $frame]]
  }

  $sel1 delete

  return [expr $sum / $max_i]
}

proc plot_average_distance {filename atoms1 atoms2 {start_frame 0}} {
  set nf [molinfo top get numframes]

  set chan [open $filename w]
  puts "started plot_average_distance: [clock format [clock scan now]]"
  for {set i $start_frame} {$i < $nf} {incr i} {
    puts $chan "$i [average_distance $atoms1 $atoms2 $i]"
  }
  puts "ended: [clock format [clock scan now]]"
  close $chan
}

proc plot_box_x {filename {start_frame 0} {x_dim 0}} {
  set nf [molinfo top get numframes]

  set chan [open $filename w]

  set box [pbc get -all]
  for {set i $start_frame} {$i < $nf} {incr i} {
    puts $chan "$i [lindex $box $i $x_dim]"
  }

  close $chan
}
```

8.2.3 write_binding_lipids.tcl

```
puts "example:
write_binding_lipids \"protein and name CA\" \"resname TLCL and name C2\"
↪   \"binding_lipids/bl\""

proc write_binding_lipids {atoms1 atoms2 filestart {startframe 0} {file_end
↪   ""} {radius 8}} {
  set nf [molinfo top get numframes]

  puts "starting from frame ${startframe}, using file ending $file_end and
↪   radius $radius"

  set sel1 [atomselect top $atoms1]
```

```
  set idx1 [$sel1 list]
  set max_i [llength $idx1]
  set chan 0

  for {set i 0} {$i < $max_i} {incr i} {
    set index [lindex $idx1 $i]
    set found 0
    set atom [atomselect top "index $index"]
    set filename "${filestart}_[$atom get segname]_[$atom get resid]_[$atom
↪   get resname]${file_end}.txt"

    for {set j $startframe} {$j < $nf} {incr j} {
      set within [atomselect top "$atoms2 and (same residue as ((same residue
↪   as ($atoms2)) and within $radius of (same residue as index $index)))"
↪   frame $j]

      if {[$within num] > 0} {
        if {$found == 0} {
          set found 1
          set chan [open $filename w]
        }

        puts $chan "$j [$within get resid]"
      }
      $within delete
    }
    puts "finished with $filename"
    $atom delete
    if {$found == 1} {
      close $chan
    }
  }

  $sel1 delete
}

# read the binding lipids for all residues in one frame before moving to next
↪   frame (maybe slightly faster)
proc write_binding_lipids2 {atoms1 atoms2 filestart {startframe 0} {file_end
↪   ""} {radius 8}} {
  set nf [molinfo top get numframes]

  set sel1 [atomselect top $atoms1]
  set idx1 [$sel1 list]
  set max_i [llength $idx1]

  set channels [lrepeat $max_i 0]
  set filenames [lrepeat $max_i ""]
  set found_values [lrepeat $max_i 0]
  for {set i 0} {$i < $max_i} {incr i} {
    set index [lindex $idx1 $i]
    set atom [atomselect top "index $index"]
    lset filenames $i "${filestart}_[$atom get segname]_[$atom get
↪   resid]_[$atom get resname]${file_end}.txt"
```

```
    $atom delete
  }

  set lines_of_each_file [lrepeat $max_i [lrepeat $nf ""]]
  set which_line_we_are [lrepeat $max_i 0]

  for {set j $startframe} {$j < $nf} {incr j} {
    for {set i 0} {$i < $max_i} {incr i} {
      set index [lindex $idx1 $i]
      set within [atomselect top "$atoms2 and (same residue as ((same residue
↪   as ($atoms2)) and within $radius of (same residue as index $index)))"
↪   frame $j]

      if {[$within num] > 0} {
        if {[lindex $found_values $i] == 0} {
          lset found_values $i 1
        }

        lset lines_of_each_file $i [lindex $which_line_we_are $i] "$j [$within
↪   get resid]"
        lset which_line_we_are $i [expr [lindex $which_line_we_are $i] + 1]
      }
      $within delete
    }
  }

  for {set i 0} {$i < $max_i} {incr i} {
    if {[lindex $found_values $i] == 1} {
      lset channels $i [open [lindex $filenames $i] w]
      for {set j $startframe} {$j < [lindex $which_line_we_are $i]} {incr j} {
        puts [lindex $channels $i] [lindex $lines_of_each_file $i $j]
      }
      close [lindex $channels $i]
    }
  }

  $sel1 delete
}
```

number_of_lipids.sh

```
#!/bin/bash

declare -a lipids

list_of_residues=$1

step_max=0
for file in *.txt ; do
  temp=${file#*_}
  id=${temp%.txt}
  last=${id##*_}
  if [[ "$last" =~ [0-9]+ ]]; then
```

```
    id=${id%_*}
  fi
  if [ "$list_of_residues" != "" ] ; then # checking the protein only
↪   partially
    found=0
    while read line ; do
      if [ "$line" == "$id" ] ; then
        found=1
        break
      fi
    done < $list_of_residues
    if [ "$found" == 0 ] ; then
      continue
    fi
  fi
  while read line ; do
    parts=($line)
    if [ ${parts[0]} -gt $step_max ]; then
      step_max=${parts[0]}
    fi
    first=1
    if [ ${lipids[${parts[0]}]+_} ]; then
      lipids_of_step=(${lipids[${parts[0]}]})
    else
      lipids[${parts[0]}]=""
      lipids_of_step=("")
    fi
    for K in ${parts[@]} ; do
      if [ $first -eq 1 ] ; then first=0; continue; fi
      found=0
      for L in ${lipids_of_step[@]} ; do
        if [ "$K" == "$L" ]; then
          found=1
          break
        fi
      done
      if [ $found -eq 0 ] ; then
        lipids[${parts[0]}]="$K ${lipids[${parts[0]}]}"
      fi
    done
  done < $file
done

step=0
found=1
while [ $step -lt $step_max ] ; do
  found=0
  lps=(${lipids[$step]})
  num=${#lps[@]}
  echo "$step $num"
  step=$(( step + 1 ))
done
```