



**Politecnico
di Torino**

Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Biomedica

A.a. 2023/2024

Sessione di Laurea Dicembre 2024

**Computational search for inhibitors of
SOD1 mutant infectivity as potential
therapeutics for ALS disease**

Relatori:

TUSZYNSKI JACEK ADAM
DERIU MARCO AGOSTINO

Candidato:

Marco Carnaroli

Abstract

Familial amyotrophic lateral sclerosis (ALS) is a progressive neurodegenerative disease characterized by the selective degeneration of motor neurons. Among the main genetic causes of ALS, over 200 mutations have been identified in the Cu/Zn superoxide dismutase (SOD1) protein, a dimeric metalloenzyme essential for converting superoxides from cellular respiration into less toxic products. Point mutations in SOD1 monomers can induce protein misfolding, which spreads to wild-type monomers through a prion-like mechanism, leading to dysfunctions that contribute to the development of the disease. Understanding the structural and functional differences between the wild-type protein and its mutated variants, as well as developing drugs capable of inhibiting the propagation of misfolding, is crucial for identifying new therapeutic strategies. In this work, seven SOD1 mutations (A4V, G41D, G41S, D76V, G85R, G93A and I104F) were selected, and three-dimensional models of SOD1 dimers composed of one wild-type monomer and one mutated monomer were generated, along with a control dimer consisting solely of wild-type monomers. Molecular dynamics simulations were conducted to investigate conformational differences between the dimers. Additionally, molecular docking was performed using a library of ligands to identify compounds with high affinity for the mutated dimers. The study reveals some differences in the mutated dimers following MD simulations and in the docking of the selected ligands with the various dimers. It provides a solid foundation for future research, paving the way for the inclusion of additional mutations and ligands to better understand the genetic causes of ALS and to develop innovative therapies.

Acknowledgements

I would like to express my gratitude to my supervisor, Professor Jacek Adam Tuszynski, for his guidance throughout the thesis journey, as well as for the opportunities and tools provided, which made this work possible. A sincere thank you also goes to my co-supervisor, Professor Marco Agostino Deriu. I am deeply grateful to Paola Vottero for her invaluable support in helping me become familiar with Canadian clusters, which were essential for conducting molecular dynamics simulations. My thanks also extend to Professor Michael Woodside for the valuable information he provided.

Ringraziamenti

Desidero innanzitutto esprimere la mia profonda gratitudine alla mia famiglia, che mi ha accompagnato lungo tutto il percorso magistrale, sostenendomi anche a distanza. Un ringraziamento speciale va ai miei genitori, Alessandro e Lucia, ai miei nonni Tonino e Anna, ai miei zii Francesca, Massimo, Alessandra e Paolo, Pina, a mio cugino Filippo, a Gianluca e Karina, e a coloro che non ci sono più: zia Paola e i nonni Dino e Tina. La loro vicinanza è stata un sostegno fondamentale nei momenti più difficili, aiutandomi a superare le sfide accademiche e personali.

Un grazie di cuore ai colleghi universitari incontrati lungo il percorso, che sono stati preziosi compagni di viaggio, amici sinceri e fonti di confronto. Un pensiero speciale va al coro universitario del Politecnico, il Polietnico, che mi ha accolto con calore, regalandomi esperienze indimenticabili dentro e fuori Torino. Quei momenti di svago e leggerezza, lontano dagli impegni accademici, rimarranno sempre nel mio cuore. Un ringraziamento particolare va anche al coro Cantabilab, per la sua inclusività e per i bei momenti condivisi.

Ringrazio di cuore i miei coinquilini Matteo e Giuseppe, con cui ho condiviso una convivenza serena e la proprietaria dell'appartamento per la sua disponibilità e gentilezza.

Un pensiero affettuoso va ai miei amici di Fano, che non mi hanno mai fatto mancare la loro presenza, anche nei momenti più difficili. Le risate, le uscite, le feste e i tanti bei momenti passati insieme sono stati una vera fonte di gioia e supporto.

Infine, il mio ringraziamento va alla città di Torino, che mi ha accolto con calore, e al Politecnico di Torino, per avermi offerto un ambiente stimolante in cui crescere e ampliare le mie conoscenze.

Table of Contents

| | |
|--|-----------|
| INTRODUCTION | 1 |
| AMYOTROPHIC LATERAL SCLEROSIS (ALS) | 2 |
| <i>Brief history</i> | 2 |
| <i>Causes and risk factors</i> | 2 |
| <i>Symptoms and phenotypes</i> | 3 |
| <i>Progression and life span</i> | 3 |
| PROTEINS | 5 |
| <i>Definition, primary structure and introduction of the folding problem</i> | 5 |
| <i>Protein building block: amino acids</i> | 5 |
| Main characteristics..... | 5 |
| Classification criteria | 7 |
| Chemistry..... | 10 |
| <i>Influencing forces in protein structure</i> | 13 |
| Hydrophobic Interactions..... | 13 |
| Van der Waals Interactions | 14 |
| Hydrogen Bonds..... | 15 |
| Electrostatic interactions | 16 |
| <i>Protein structure</i> | 17 |
| Secondary structure | 17 |
| α -helices | 18 |
| β -structure: strands and pleated sheets..... | 18 |
| Beta hairpin connection (β -motif)..... | 21 |
| Crossover Connection ($\alpha\beta$ -motif)..... | 21 |
| Tight turns | 22 |
| β -bulge | 22 |
| Disulfide bridges..... | 23 |
| Random coil | 23 |
| β -barrels | 23 |
| Tertiary structure..... | 24 |
| Structure based protein classification: fibrous and globular | 24 |
| SOD PROTEIN FAMILY | 26 |
| <i>Function and importance</i> | 26 |
| <i>Dismutation reaction</i> | 26 |
| SOD1 | 27 |
| <i>SOD1 dismutation reaction</i> | 27 |
| <i>SOD1 structure</i> | 29 |
| <i>Description of SOD1 dismutation reaction based on structure</i> | 31 |
| <i>SOD1 folding and maturation</i> | 31 |
| SOD1 MUTATIONS IN ALS | 32 |
| <i>From loss of function to toxic acquired behaviour</i> | 32 |
| <i>Prion like mechanism of conversion of SOD1</i> | 33 |
| AIM OF THE WORK | 36 |
| <i>Proposed mechanism for prion like conversion inhibition</i> | 36 |
| METHODS | 38 |
| OBTAINING THE STRUCTURE OF THE HUMAN WILD-TYPE HOLO PROTEIN | 38 |
| GENERATION OF THE MUTANT MONOMERS | 38 |
| <i>Generation of the mutated sequences</i> | 39 |
| <i>Obtaining the mutated monomeric structures in pdb format</i> | 39 |
| <i>Obtaining the 'holo' variants of the mutated monomeric structures</i> | 39 |
| GENERATION OF THE DIMERS | 40 |
| MD SIMULATIONS | 40 |
| <i>Dimers preparation and simulation run</i> | 40 |
| <i>Analysis</i> | 43 |
| CANDIDATE DRUGS: PREPARATION FOR DOCKING..... | 44 |
| VIRTUAL SCREENING WITH DOCKING PROCEDURE | 46 |
| <i>AUTODOCK VINA blind docking</i> | 46 |
| RESULTS AND DISCUSSION | 48 |
| STRUCTURE OF THE HUMAN WILD-TYPE HOLO PROTEIN | 48 |

| | |
|---|-----------|
| MUTANT MONOMERS GENERATION..... | 48 |
| <i>Mutated sequences</i> | 48 |
| <i>Mutated structures</i> | 49 |
| DIMERS GENERATION..... | 50 |
| <i>Dimer interface characteristics</i> | 50 |
| <i>Ramachandran plots</i> | 51 |
| MD SIMULATIONS: DATA ANALYSIS | 52 |
| <i>RMSD</i> | 52 |
| <i>Radius of gyration</i> | 53 |
| <i>SASA</i> | 54 |
| <i>RMSF</i> | 55 |
| <i>Conformational clustering</i> | 57 |
| <i>Ramachandran plots</i> | 58 |
| VIRTUAL SCREENING WITH DOCKING PROCEDURE | 59 |
| <i>AUTODOCK VINA blinded docking, affinity evaluation and pose clustering</i> | 59 |
| CONCLUSION | 64 |
| FUTURE RESEARCH..... | 64 |
| REFERENCES..... | 65 |
| APPENDIX 1: SOFTWARES..... | 69 |
| APPENDIX 2: SCRIPTS | 71 |

Introduction

The study aims to conduct a computational search for drugs capable of inhibiting the prion-like infection mechanism of SOD1 mutants, which are at the basis of ALS. ALS is a severe and progressive neurodegenerative disease, characterized by the selective loss of motor neurons in the central nervous system, leading to impairment of motor function, paralysis, and ultimately death. In its familial form, several genes have been identified as possible causes; the first is SOD1, which shares its name with the dimeric protein it encodes. It is hypothesized that, in its mutated form, this protein may destabilize and subsequently induce destabilization of the wild-type protein, through a prion-like mechanism. Although native SOD1 consists of two identical monomers, whose dual presence enhances enzymatic activity, it may happen that monomers with a point mutation are synthesized. These mutant monomers, following a mechanism similar to that of prions, would transfer their conformational characteristics to wild-type monomers when in contact with them, inducing the latter to adopt an incorrect, destabilizing conformation. The dimer thus formed, composed of a mutant monomer and an “infected” wild-type monomer, would dissociate, allowing the mutant monomer to “infect” other wild-type monomers, thereby spreading the misfolded conformation of the protein without the intervention of foreign agents, just like a prion. Ligands with the best affinity for the dimers were thus sought.

The study is divided into three phases, preceded by a review of the disease and the protein under investigation. In the first phase, protein structures in PDB format were obtained for further study. Since mutated monomer structures were unavailable, they were predicted using AlphaFold2. However, the resulting predicted structures lacked metal ions, which are essential for enzymatic activity and, above all, protein stability; these ions were manually added with the help of MOE software. The monomers were then combined using Haddock software, resulting in eight dimers, including the completely wild-type protein, namely: wt+wt, wt+A4V, wt+G41D, wt+G41S, wt+D76V, wt+G85R, wt+G93A, and wt+I104F.

In the second phase, after careful preparation, 1200 ns MD simulations were performed using Amber software on the obtained structures. Following the simulations, a series of analyses and conformational clustering were carried out to highlight the differences between the mutated and wild-type structures.

In the third phase, 35 ligands were selected and, using VINA software, a blind docking (since no information on the protein’s specific binding site was available) was conducted for each ligand on every structure, both mutated and not, for a certain number of iterations. This allowed data collection on binding affinity, or binding energy, between each ligand-protein pair. Finally, clustering was performed on the poses obtained for each ligand, to determine if each ligand tended to bind recurrently to a specific protein region, thereby providing insights into the ligand’s specificity.

Amyotrophic lateral sclerosis (ALS)

Brief history

Discovery of Amyotrophic lateral sclerosis (ALS) is associated with the name of Jean-Martin Charcot (1825-1893), first professor of neurology at the Salpêtrière. However, there are some earlier descriptions ALS related, such as a progressive muscular weakness accompanied by muscle wasting clearly recognized by the mid-19th century. In 1848 a syndrome of this kind was described by Francois Aran, who correctly suspected a neurogenic disease. After having studied 11 patients with the same symptoms, he suggested that that was a new type of syndrome, called progressive muscular atrophy (PMA). One of his colleagues, Duchenne, at first considered PMA as myogenic, but, due to changes in the spinal cord found by Cruveilhier, it was concluded that the cause was neurogenic. Even though Charcot and his colleague Joffroy described two cases of PMA with lesions in the posterolateral spinal cord without naming it explicitly ALS, in 1874 Charcot recognized ALS as a distinct syndrome. His description of ALS was derived from observations of 20 patients and five autopsy examinations. Since Salpêtrière was a women's hospital, most of these patients were women. Since then, clinical, and pathological characteristics of ALS has been modified very little [1].

Causes and risk factors

Two main types of this syndrome have been discovered, sporadic and familial, whose incidence and main causes are shown in figure 1.

Sporadic ALS is the most common form, accounting for 90% to 95% of all cases of ALS it can affect anyone, and it occurs randomly without any known genetic cause or family history of ALS even though can occasionally be associated with gene changes (mutations) [2]. In sporadic ALS, risk factors include genetic, environmental and lifestyle.

Familial or genetic ALS, instead, is an inherited form of the disease that affects a small amount of people. In families with genetic ALS, each child has up to a 50-60% chance of inheriting the gene and developing the disease [3].

Since 1993, large scale genetic studies have identified more than 60 genes that are associated with ALS. C9ORF72, SOD1, FUS and TARDBP are the most common causative genes, accounting for more than 50% of familial ALS (FALS) but also approximately 7.5 % of sporadic ALS. However, several genes associated with ALS are risk genes [1].

Environmental, lifestyle and occupational risk factors are important, but their identification is hard. Over several decades, a variety of non-genetic risk factors for ALS have been examined, yet only a handful have met the stringent criteria set by epidemiologists. Consequently, definitive non-genetic risk factors for ALS that are replicable across studies have not been established [1].

Head injuries, physical activity, exposure to electric shock, military service, pesticides, lead, and other factors have been regarded as possible risk factors for both the onset and progression of ALS [1]. A new study published in JAMA Network Open suggests that professional football may be a risk factor for ALS. Between 1960 and 2019, out of 19,423 NFL players, 38 were diagnosed with ALS and 28 died from it. The risk of developing and dying from ALS among these players was nearly four times higher than that of the general male population. Players with ALS had an average career length of seven years, compared to 4.5 years for those without the disease. Many were diagnosed in their mid-30s, younger than the typical age for ALS onset. The study did not explore the reasons behind this link, but the authors suggest traumatic brain injury might be a contributing factor [4], [5]. It was also found possible independent associations between occupational exposure to extremely low-frequency magnetic fields (ELF-MF) and electric shocks with the risk of developing ALS [6].

However, to date, advanced age and male gender are the only established non-genetic risk factors for ALS [1].

Symptoms and phenotypes

ALS is a complex neurodegenerative disease that affects humans exclusively. It can impact any muscle in the body, leading to varied effects in each individual. For example, some individuals may develop a form of ALS that predominantly impacts their respiratory muscles. In such cases, they experience shortness of breath, even with minimal physical activity. For others, ALS may initially target the muscles in the throat and mouth, leading to swallowing difficulties, a condition often known as bulbar ALS. Additionally, the disease progresses at different rates for different people. In some, it advances rapidly, while in others, it progresses more slowly. Common symptoms include painless, progressive muscle weakness, which may cause someone to trip more often or drop things, other symptoms can include slurred speech, difficulty swallowing, and trouble breathing. In Table 1 some of the symptoms can be visualized, and they with the related frequency in terms of number of patients for each type of ALS [3]. In Table 1, the symptoms and the number of patients for each type of ALS, as described by the study of Chiò et al., are shown [7].

Table 1: The patients described by Chiò et al. [7].

| ALS type | Symptoms | No. of people with this type |
|---|---|------------------------------|
| Bulbar | Affects clarity of speech or ability to swallow | 999 |
| Classic | Can affect arms or legs | 835 |
| Predominantly upper motor neuron | Paraplegia or quadriplegia | 240 |
| Flail arm | Shoulder (upper arm) weakness for >12 months | 187 |
| Flail leg | Lower leg weakness for >12 months | 531 |
| Respiratory | Difficulty breathing | 47 |

Abbreviation: ALS = amyotrophic lateral sclerosis.

Several phenotypes contribute to defining ALS, a uniquely human neurodegenerative disease, such as age of onset, site of initial clinical presentation, disease duration, which are based on genetic, environmental, lifestyle and epigenetic causes. Sadly, there are no animals manifesting this kind of disease and induced animal models can't summarize all shapes of the disorder as seen in humans. The shortcomings of ALS animal models are likely due to multiple factors, including major anatomical differences, especially those related to the corticomotoneuronal system. Furthermore, variations in aging and lifespan, lifestyle factors, and the wide range of environmental exposures that humans encounter are also significant contributors. ALS is now increasingly studied using patient-derived induced pluripotent stem cells (iPSCs), which are generated from skin or blood cells that have been reprogrammed to return to an embryonic-like pluripotent state. This approach also offers the potential advantage of investigating the early stages of ALS [1].

In the majority of cases, ALS begins in patients around the age of 60, manifesting as painless, asymmetrical weakness in a limb, which is known as spinal-onset ALS. In approximately 20% of ALS patients, the initial weakness affects the bulbar muscles, leading to symptoms such as dysarthria, dysphagia, and tongue fasciculations. In roughly 3–5% of patients, ALS begins with respiratory symptoms such as orthopnoea or dyspnoea, with minimal or no spinal or bulbar signs. Respiratory-onset ALS is more common in males and has a particularly poor prognosis, with an average survival time of 1.4 years and no cases of long-term survival. While ALS typically begins in the fifth or sixth decade of life, it can occur at almost any age. Juvenile ALS, defined as having an onset before the age of 25, generally progresses more slowly than other forms of the disease [8].

Progression and life span

Although the median survival in ALS is generally around three years from diagnosis, there is remarkable variability in survival, which reflects the variability in the rate of disease progression. At one end of the spectrum are the 10% of ALS patients who live longer than 10 years. Autopsies show

their disease is indistinguishable from classic ALS. Long survival is more common in patients with juvenile ALS and upper motor neuron-predominant ALS. Certain hereditary forms of ALS are linked to particularly long or short survival times. For instance, the SOD1 Asp90Ala mutation is associated with long survival, while the SOD1 Ala4Val mutation can cause a very aggressive form of the disease. Even among family members with the same mutation, there is significant variability in progression rates. This suggests that the relationship between the genetic cause and the phenotype is complex, with genetic or environmental factors influencing phenotypic expression, particularly age at onset and disease progression. Identifying these modifying factors is crucial, as they could serve as targets for therapeutic intervention, even without understanding the disease's cause. Delaying the age of onset or slowing the progression rate and consequent functional decline is of clear therapeutic interest. Several small animal models, such as *Drosophila* and zebrafish, have been used to screen for such modifiers. This research has led to the identification of druggable targets and even novel causes of ALS [8].

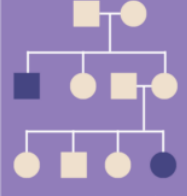
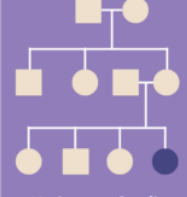
| | Definition | Prevalence | Causes |
|--------------|---|------------------------------|--|
| Familial ALS |  <p>At least one family member with ALS</p> | Roughly 10% of all ALS cases | About 60% of cases have a known disease-causing mutation. |
| Sporadic ALS |  <p>No known family history of ALS</p> | Around 90% of all ALS cases | Environmental and lifestyle factors play a larger role. Only 10% of cases have a known ALS-related mutation. |

Figure 1: Main types of ALS and their characteristics [9].

Proteins

Definition, primary structure and introduction of the folding problem

Proteins are large biomolecules and macromolecules made up of one or more long chains of amino acid residues, encoded by combinations of three DNA building blocks (nucleotides), determined by the sequence of genes [10], [11]. Our body can synthesize some amino acids but essential amino acids cannot be produced by the body and must be obtained from dietary sources [12]. A wide variety of functions is performed by proteins within organisms, such as catalyzing metabolic reactions, replicating DNA, responding to stimuli, providing structure to cells and organisms, and transporting molecules from one place to another. The primary difference between proteins lies in their amino acid sequence, which is determined by the nucleotide sequence of their genes. A linear chain of amino acid residues is called a polypeptide, and a protein contains at least one long polypeptide. Short polypeptides, with fewer than 20–30 residues, are seldom considered proteins and are commonly referred to as peptides. The sequence of amino acid residues in a protein is defined by the sequence of a gene encoded in the genetic code, which generally specifies 20 standard amino acids [10]. Proteins, even though they have a specific primary structure which is basically defined by their aminoacidic sequence, usually exist as compact, folded structures rather than fully extended polypeptide chains. In fact, the function of a protein is rarely determined solely by its amino acid sequence, even if this one is the main component. Instead, its ability to function is typically determined by its overall three-dimensional shape or conformation. This native, folded structure is influenced by several factors:

- (1) interactions with solvent molecules such as water.
- (2) the pH and ionic composition of the solvent.
- (3) the sequence of the protein, which is the most important.

The third factor is the most unpredictable: the primary structure allows for both close interactions between neighboring segments and distant interactions between parts of the sequence that are farther apart. While the overall structure of the protein might seem disordered and arbitrary at first glance, it almost always reflects a precise and complex equilibrium of various forces that together establish the protein's unique shape. [13].

Protein building block: amino acids

Main characteristics

As previously told, amino acids are the building blocks of proteins, being part their primary structure. There are more than 300 different amino acids described in nature, although only 20 are commonly found in mammalian and so human proteins. Each amino acid has a carboxyl group ($COOH$), a primary amino group (NH_2), a hydrogen atom (H) and a unique side chain (“R-group”) covalently bonded to the tetrahedral α -carbon atom. It is the nature of the distinctive side chain (different for each amino acids) that regulates the role each amino acid plays in a protein and gives each one its identity [14], [15]. In neutral solution (pH 7), the carboxyl group exists as $-COO^-$ while the primary amino group as $-NH_3^+$, so, since the resulting amino acid contains one positive and one negative charge, it is called zwitterion, basically a neutral molecule. The structure of a generic amino acid can be easily visualized in figure 2.

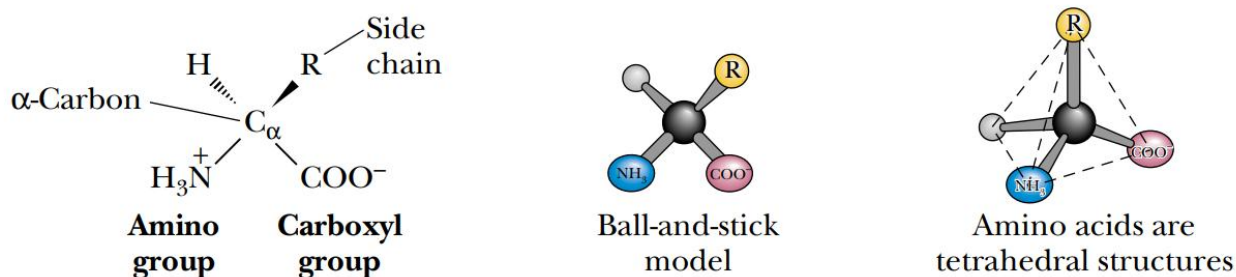


Figure 2: triple representation of a generic amino acid, [15].

Except for glycine, all other 19 amino acids are chiral molecules. This is because they have four different groups bonded to the α -carbon center, which is said to be asymmetric [15]. That means that they are not superimposable with their mirror image and so they are called enantiomers or optical isomers (fig. 3). Those are part of a wider category of isomers, stereoisomers, which share the same molecular formula and bonded atom sequence in their structures but differ solely in their spatial arrangement of bonded atoms. Only glycine, containing hydrogen as its R-group does not have chirality. Despite their nearly identical physical properties enantiomers do differ in two notable ways: their ability to rotate plane-polarized light and their interactions with biological molecules such as receptors and enzymes while those that rotate light to the right are termed dextrorotary and are denoted with a D-letter designation (fig. 2). Those that rotate light to the left are called levorotary and are marked with an L-letter designation, helping to differentiate between the two enantiomers [16]. In proteins almost all amino acids, carboxyl $-COO^-$ and amino $-NH_3^+$, groups are linked in a peptide bond through a condensation reaction, precluding any other chemical reaction except for hydrogen bond formation. In figure 3 the amino and carboxyl groups react in a head-tail fashion, eliminating a water molecule and forming a covalent amide linkage, known as peptide bond (fig. 4)). In aqueous solutions, the equilibrium of this reaction tends to favor the hydrolysis of peptide bonds [15].

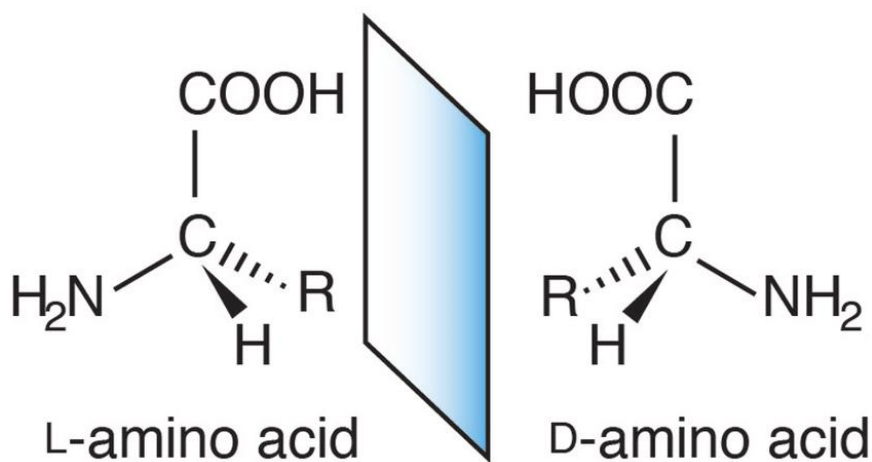


Figure 3: example of a generic amino acid enantiomer, with L and D forms [17].

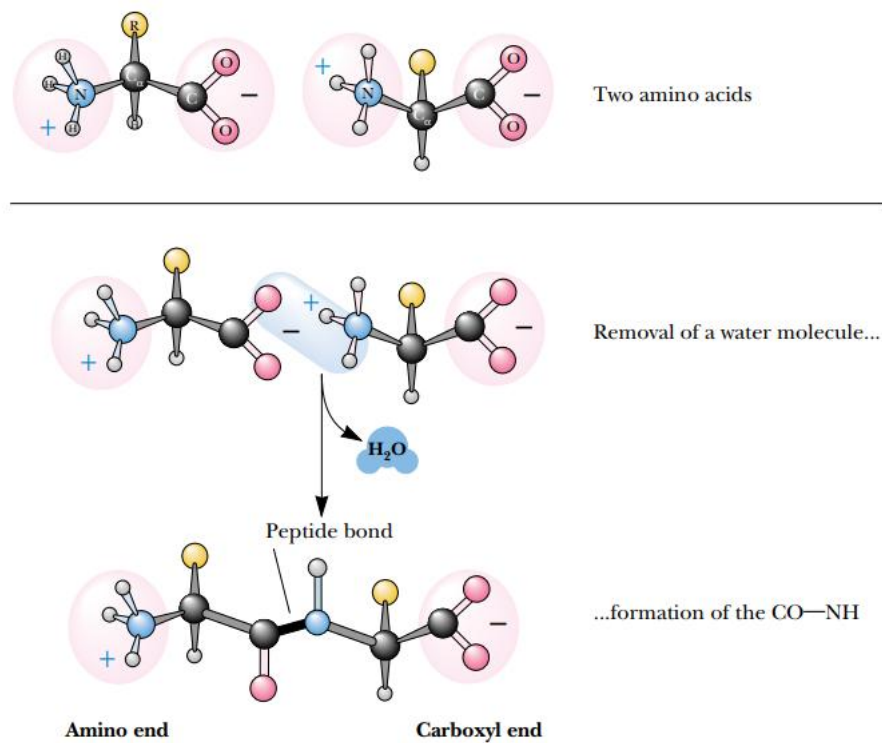


Figure 4: peptide bond formation, [14].

Classification criteria

There are various methods to categorize common amino acids, but the most effective classification system is based on the polarity of their side chains (R-group). So, following this criterion, amino acids can be grouped in four categories as shown in table 2 and their structure with a triple representation can be seen in figure 5.

Table 2: Amino acids classification based on polarity.

| Nonpolar | Polar | | |
|--|-----------------------------------|---|------------------------------------|
| alanine, valine, leucine, isoleucine, proline, methionine, phenylalanine, tryptophan | Basic (positively charged) | Neutral | Acidic (negatively charged) |
| | histidine, arginine, and lysine | glycine, serine, threonine, asparagine, glutamine, cysteine, tyrosine | aspartic acid and glutamic acid |

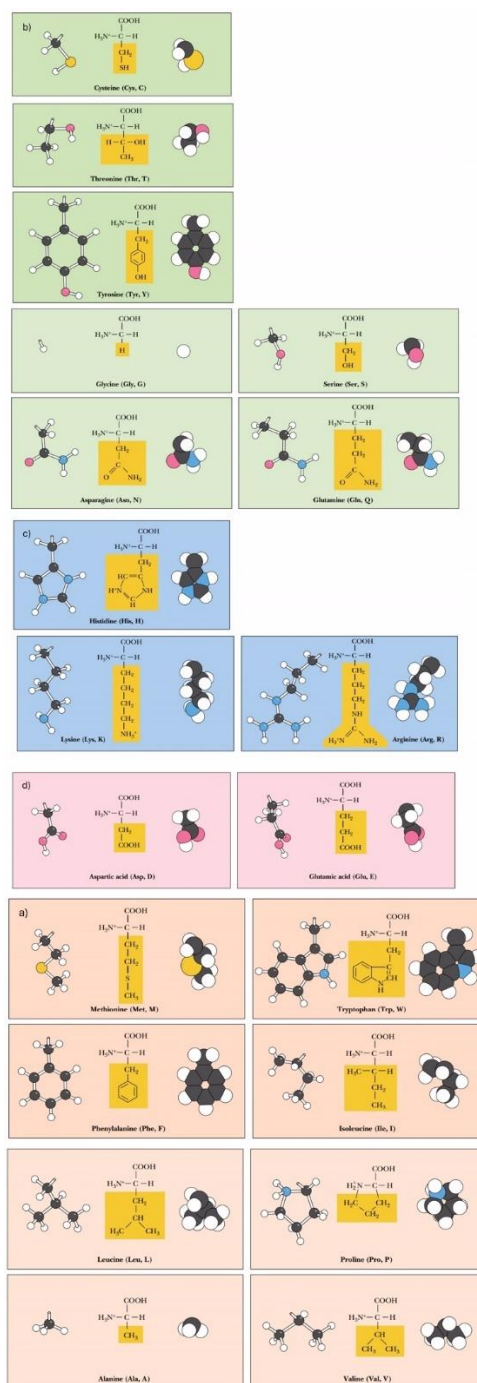


Figure 5: representations of amino acids differentiated by category; a) Nonpolar (hydrophobic), b) Polar, uncharged, c) Basic, d) Acidic [15].

Nonpolar amino acids consist of those with alkyl chain R groups, such as alanine, valine, leucine, and isoleucine. This group also includes proline, known for its unique cyclic structure, methionine, one of the two sulfur-containing amino acids, and the aromatic amino acids phenylalanine and tryptophan. Polar, uncharged amino acids, include glycine, serine, threonine, asparagine, glutamine, cysteine, tyrosine. Except for glycine, they have R groups capable of forming hydrogen bonds with water, making them generally more soluble in water than their nonpolar counterparts. However, there are notable exceptions. Tyrosine has the lowest water solubility among the 20 common amino acids. Proline is highly water-soluble, while alanine and valine have solubility comparable to arginine and serine. Asparagine and glutamine's amide groups, the hydroxyl groups of tyrosine, threonine, and

serine, and cysteine's sulfhydryl group are all proficient at forming hydrogen bonds. Glycine, the simplest amino acid, has only a single hydrogen as its R group, which is not effective at hydrogen bonding. Glycine's solubility is primarily due to its polar amino and carboxyl groups, classifying it as a polar, uncharged amino acid. Tyrosine, despite its significant nonpolar characteristics from its aromatic ring, has a phenolic hydroxyl group with a pKa of 10.1 which is a charged, polar entity at high pH and so tyrosine is part of the polar group.

Only two acidic amino acids —aspartic acid and glutamic acid— exist and their side chains contain a carboxyl group. These carboxyl groups are weaker acids than the α -COOH group but are sufficiently acidic to exist predominantly in the form of $-COO^-$ at neutral pH. Therefore, at pH 7, both aspartic acid and glutamic acid carry a net negative charge. These negatively charged amino acids fulfill several crucial roles in proteins. Aspartic acid and glutamic acid, thanks to their side chains, are present in a lot of proteins that bind metal ions for structural or functional purposes. Furthermore, carboxyl groups can act as nucleophiles in specific enzyme reactions and participate in various electrostatic bonding interactions.

Three common amino acids possess side chains that are positively charged at neutral pH: histidine, arginine, and lysine. Those are called basic amino acids. Histidine's ionized group is an imidazolium, arginine's is a guanidinium group, and lysine contains a protonated alkyl amino group. At pH 7, the side chains of arginine and lysine are fully protonated, while histidine, with a side chain pKa of 6.0, is about 10% protonated at pH 7. Due to its near-neutral pKa, histidine side chains play crucial roles as both proton donors and acceptors in numerous enzyme reactions. Arginine and lysine side chains, being protonated under physiological conditions, contribute to electrostatic interactions within proteins [15].

Another less common way to classify amino acids is based merely on the R-group characteristics [18]. They can be grouped in 10 categories that can be visualized in table 3.

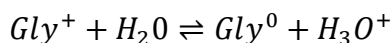
Table 3: Amino acid classification based solely on R-group features [18].

| | | | | | |
|--------------|--|--|------------------------------------|----------------------------------|---|
| Class | Simple | Hydroxy | Sulfur containing | Aromatic | Heterocyclic |
| Features | No functional group | Hydroxyl group on their side chain | Sulfur in their side chain | Benzene ring in their side chain | side chain ring with at least an atom other than carbon |
| Components | glycine, valine, alanine, leucine, isoleucine | serine, threonine | cysteine, methionine | phenylalanine, tyrosine | tryptophan, histidine, proline |
| Class | Amine group containing | Branched chain | Acidic | Basic | Imino |
| Features | derivatives of amino acids in which one of carboxyl group has been transformed into an amide group | A branched-chain amino acid (BCAA) is an amino acid having aliphatic side-chains with a branch | carboxyl group in their side chain | amino group in their side chain | Amino acids containing a secondary amine group |
| Components | asparagine, glutamine | leucine, isoleucine, valine | aspartic and Glutamic acid | lysine, Arginine | proline |

Chemistry

Brønsted and Lowry define an acid as a proton donor and a base as a proton acceptor. This concept is crucial for understanding monoprotic and polyprotic acids and bases. Monoprotic substances involve the transfer of a single proton, while polyprotic substances can transfer multiple protons. Consequently, a monoprotic acid can donate only one proton, whereas a polyprotic acid can donate more than one. Likewise, a monoprotic base can accept only one proton, whereas a polyprotic base can accept multiple protons [19]. Common amino acids are characterized as weak polyprotic acids, so acids that can lose several protons per molecule. Their ionizable groups do not dissociate strongly, and the extent of dissociation depends on the pH of the environment. Each amino acid contains at least two hydrogen atoms that can be dissociated, so they are at list diprotic.

Examining the acid-base behavior of the simplest amino acid, glycine, at low pH levels, both its amino and carboxyl groups are protonated, giving the molecule a net positive charge Gly^+ . As the pH rises, the carboxyl group dissociates first, forming the neutral zwitterionic species Gly^0 . With further increases in pH, the amino group dissociates, resulting in the negatively charged glycinate form Gly^- (fig. 6). So, the first dissociation reaction that gives Gly^0 from Gly^+ can be written as follows:

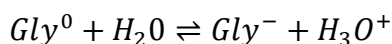


and the relative dissociation constant as:

$$K_1 = \frac{[Gly^0][H_3O^+]}{[Gly^+]}$$

Typical values for K_1 for most of the amino acids vary from 0.4 to $1.0 \times 10^{-2} M$, so that pK_1 values go from 2.0 to 2.4, knowing the conversion formula $pK_x = -\log K_x$.

The second dissociation reaction which transforms Gly^0 into Gly^- is:



and, subsequently, the dissociation constant K_2 is:

$$K_2 = \frac{[Gly^-][H_3O^+]}{[Gly^0]}$$

pK_2 values are in the range of 9.0 to 9.8.

At physiological pH, the α -carboxyl group of a simple amino acid (without ionizable side chains) is fully dissociated, while the α -amino group remains largely undissociated. The titration curve for glycine is shown in figure 7.

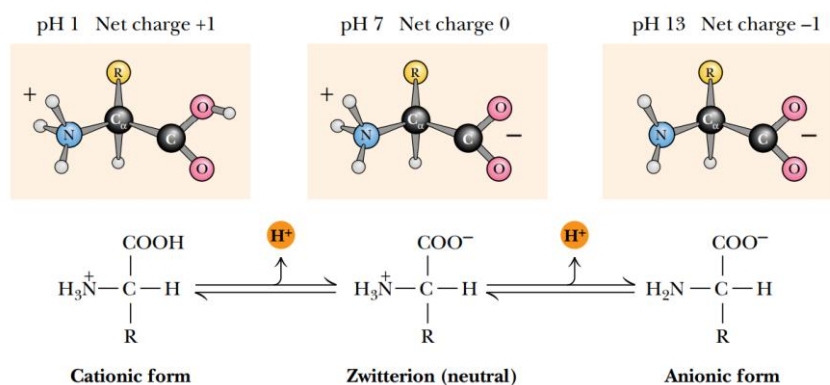


Figure 6: Ionic forms of amino acids, without taking into consideration the ionizable nature of the R-group that constitutes the side chain [15].

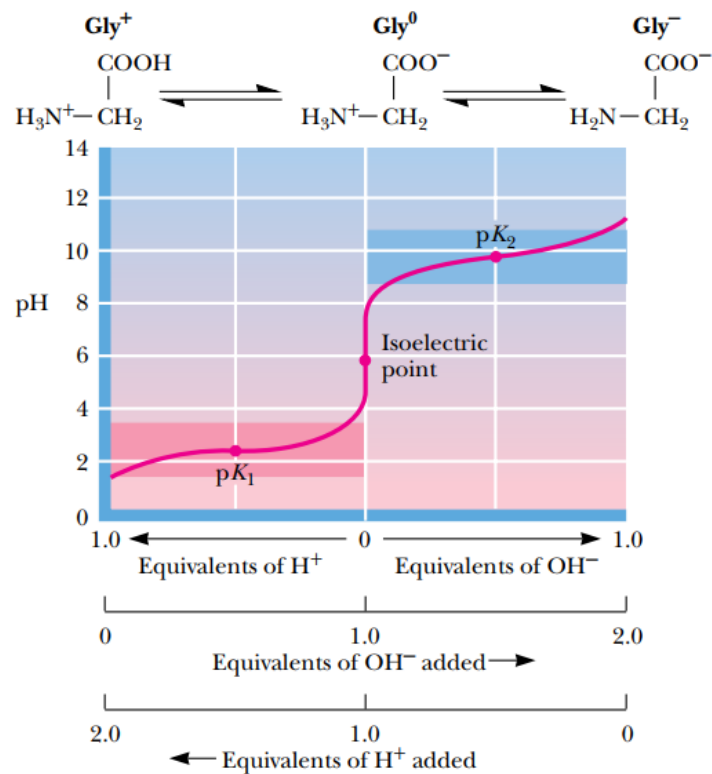


Figure 7: Titration of glycine. The pI, the pH where the molecule has a net charge of 0, is defined as average between pK_1 and pK_2 [15]

However not only α -carboxyl and α -amino groups undergo dissociation like it happens in glycine but also R-groups of several amino acids. pK_a of these groups can be viewed here in table 4.

Table 4: pK_a of each group for all 20 amino acids [15].

| pK_a Values of Common Amino Acids | | | |
|-------------------------------------|-----------------------|---|----------------|
| Amino Acid | α -COOH pK_a | α -NH ₃ ⁺ pK_a | R group pK_a |
| Alanine | 2.4 | 9.7 | |
| Arginine | 2.2 | 9.0 | 12.5 |
| Asparagine | 2.0 | 8.8 | |
| Aspartic acid | 2.1 | 9.8 | 3.9 |
| Cysteine | 1.7 | 10.8 | 8.3 |
| Glutamic acid | 2.2 | 9.7 | 4.3 |
| Glutamine | 2.2 | 9.1 | |
| Glycine | 2.3 | 9.6 | |
| Histidine | 1.8 | 9.2 | 6.0 |
| Isoleucine | 2.4 | 9.7 | |
| Leucine | 2.4 | 9.6 | |
| Lysine | 2.2 | 9.0 | 10.5 |
| Methionine | 2.3 | 9.2 | |
| Phenylalanine | 1.8 | 9.1 | |
| Proline | 2.1 | 10.6 | |
| Serine | 2.2 | 9.2 | ~13 |
| Threonine | 2.6 | 10.4 | ~13 |
| Tryptophan | 2.4 | 9.4 | |
| Tyrosine | 2.2 | 9.1 | 10.1 |
| Valine | 2.3 | 9.6 | |

Aspartic acid and glutamic acid each have an additional carboxyl group in their side chains. For aspartic acid, this is the β -carboxyl group, and for glutamic acid, it is the γ -carboxyl group. These carboxyl groups have pK_a values that fall between those of the α -carboxyl group and typical aliphatic carboxyl groups, reflecting their intermediate acidity.

Lysine has an additional aliphatic amino group in its side chain, specifically the ϵ -amino group. This ϵ -amino group has a higher pK_a compared to the α -amino group but is like that of a typical aliphatic amino group.

Histidine contains an ionizable imidazolium proton, which makes it unique among the amino acids. Arginine has a guanidinium group in its side chain. These specific functional groups have their own characteristic pK_a values, which can be found in Table 4.1.

The slightly higher pK_a values for the side-chain groups of aspartic acid, glutamic acid, and lysine, compared to their α -carbon counterparts, indicate the reduced influence of the α -carbon on groups several atoms away on the side chain [19].

Lysin and glutamic titration curves can be visualized in figure 8.

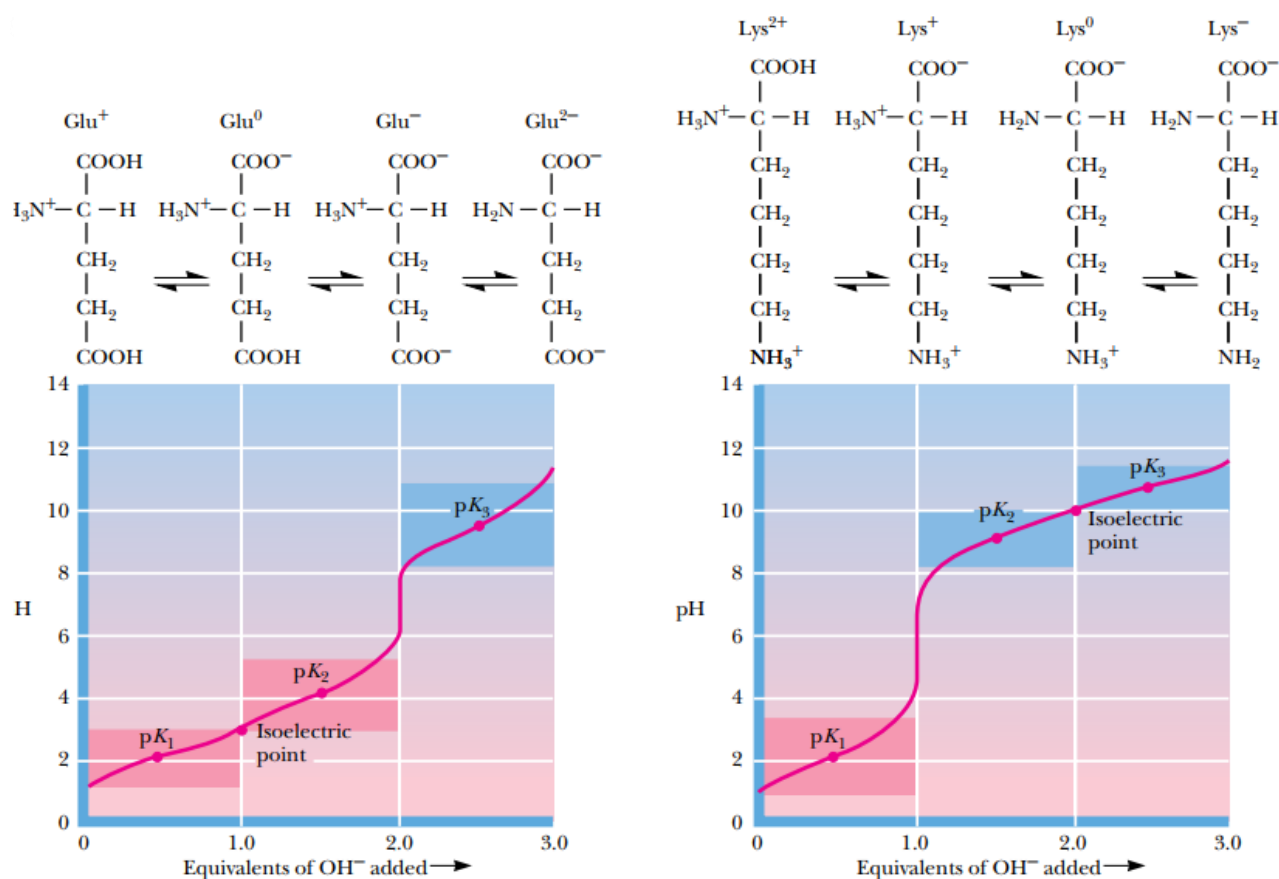


Figure 8: Titration of glutamic acid and lysine [15]

Influencing forces in protein structure

Various types of noncovalent interactions play crucial roles in protein structure. These include hydrogen bonds, hydrophobic interactions, electrostatic bonds, and van der Waals forces. Although these interactions are noncovalent, they significantly influence protein conformations [13].

Hydrophobic Interactions

Hydrophobic interactions are noncovalent forces that cause nonpolar molecules to cluster together in aqueous solutions in order to have minimal contact with water. These interactions are driven by and result in a change in the system's entropy [20].

When a hydrophobic substance, typically a nonpolar molecule with a long carbon chain, is introduced into an aqueous medium, hydrogen bonds between water molecules are disrupted to accommodate the hydrophobe. However, the water molecules do not chemically react with the hydrophobe. This process is endothermic because breaking bonds requires energy input into the system. Water molecules distorted by the hydrophobe will form new hydrogen bonds and create an ice-like structure known as a clathrate cage around the hydrophobe. This arrangement makes the system more ordered, leading to a big decrease in the overall entropy of the system, resulting in $\Delta S < 0$. Enthalpy change ΔH can be negative, zero or positive but it's negligible compared to entropy one. According to the Gibbs Energy formula:

$$\Delta G = \Delta H - T\Delta S$$

Being the entropy change really negative, the free energy change becomes positive and that indicates that the mixing of the hydrophobe and water molecules is not spontaneous.

However, interactions between hydrophobes are spontaneous, so that when hydrophobic molecules interact with each other, enthalpy increases ($\Delta H > 0$) thanks to the rupture of some hydrogen bonds that constitutes the clathrate cage. Also, the entropy will increase ($\Delta S > 0$), since forming the clathrate cage caused entropy decrease. So ΔG will become negative and hence hydrophobic interactions are spontaneous (fig. 10).

Hydrophobic interactions surpass other weak intermolecular forces like Van der Waals interactions and hydrogen bonds in strength. These interactions become stronger with an increase in temperature or the number of carbon atoms in the hydrophobic molecules.

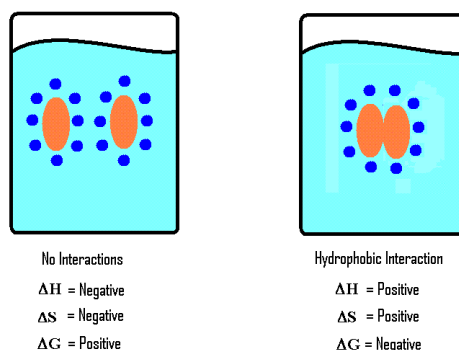


Figure 9: hydrophobic substances spontaneously aggregate together in order to expose less surface to water [21].

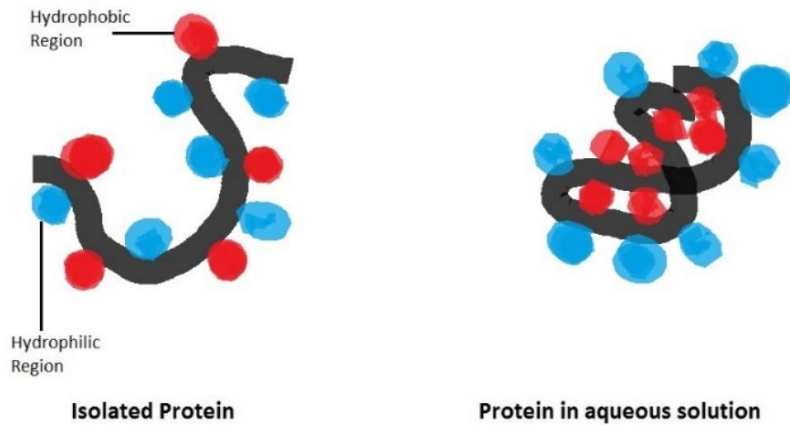


Figure 10: simplified behavior of protein folding due to hydrophobic interactions [22].

Hydrophobic interactions play a crucial role in protein folding, which is essential for maintaining a protein's stability and biological activity. These interactions help the protein reduce its surface area, minimizing unfavorable interactions with water [21].

Van der Waals Interactions

Van der Waals interactions include both attractive and repulsive forces. The attractive forces primarily arise from instantaneous dipole-induced dipole interactions, which are triggered by fluctuations in electron charge distributions among adjacent nonbonded atoms. Although individual van der Waals interactions are relatively weak, with stabilization energies between 4.0 and 1.2 kJ/mol, their collective impact in a typical protein is significant, playing a key role in the protein's stability [13]. There are two main types of Van der Waals forces: the the stronger dipole-dipole forces and the weaker London Dispersion Forces. The term "electron charge density" describes the probability of an electron being found in a particular area of the electron cloud at any moment. Electrons do not consistently occupy the same location; when they momentarily gather in one area, a temporary dipole forms, even in nonpolar molecules. This creates a partial negative charge at one end of the molecule, which induces an instantaneous dipole in surrounding molecules, attracting their positive ends. This effect is referred to as the London Dispersion Force of attraction. Dipole-dipole forces are akin to London Dispersion forces, but they occur between molecules that are permanently polar rather than momentarily polar. In these interactions, a polar molecule, like water, attracts the positive end of another polar molecule using its own negative end, resulting in a dipole-dipole force between them [22].

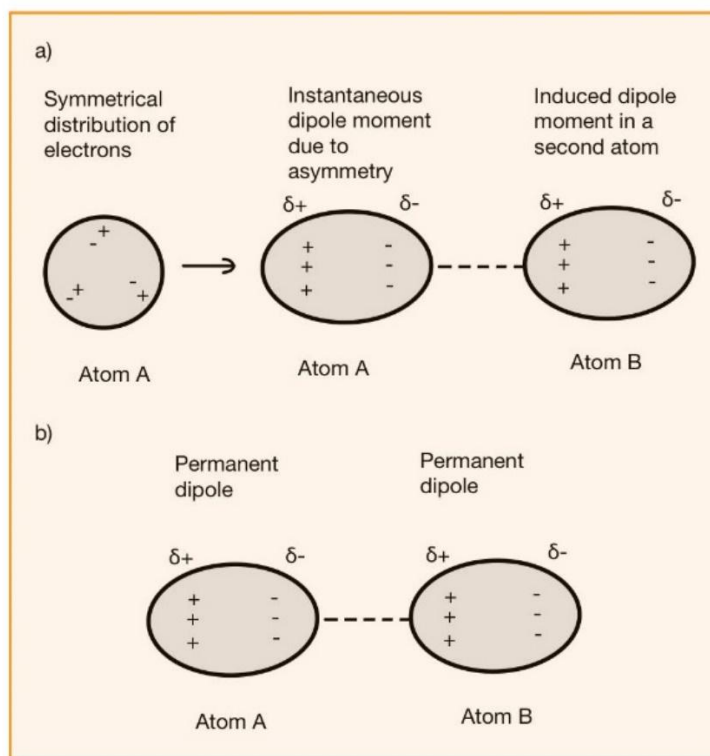


Figure 11: Dipole-dipole interactions and London interactions.

Hydrogen Bonds

Hydrogen bonds are specific types of dipole-dipole attraction that occurs when a hydrogen atom, bonded to a highly electronegative atom, is near another electronegative atom that has a lone pair of electrons. To form the bonds, a right orientation of the involved groups is needed. Those highly electronegative atoms are oxygen, nitrogen, or fluorine. In each molecule capable of hydrogen bonding, the hydrogen, being less electronegative, acquires a highly positive partial charge δ^+ while the highly electronegative atom attains a high negative partial charge δ^- , having at least one active lone pair of electrons [13], [23]. Water serves as a prime example of hydrogen bonding (fig. 6). Each water molecule can form up to four hydrogen bonds with neighboring water molecules: two through its hydrogen atoms and two through its oxygen atom. The number of δ^+ hydrogens and lone pairs in water molecules perfectly allows for all of them to participate in hydrogen bonding [23]. In the majority of analyzed protein structures, the atoms that make up the peptide backbone typically engage in hydrogen bonding with each other. Additionally, side chains that can form hydrogen bonds are generally found on the protein's surface, where they mostly bond with water molecules. Even though each hydrogen bond provides an average stabilization energy of around 12 kJ/mol, the sheer number of hydrogen bonds in a typical protein result in significant overall stabilization [13]. Hydrogen bonding plays a crucial role in the properties of biological assemblies, including membranes, proteins, and nucleic acids [24].

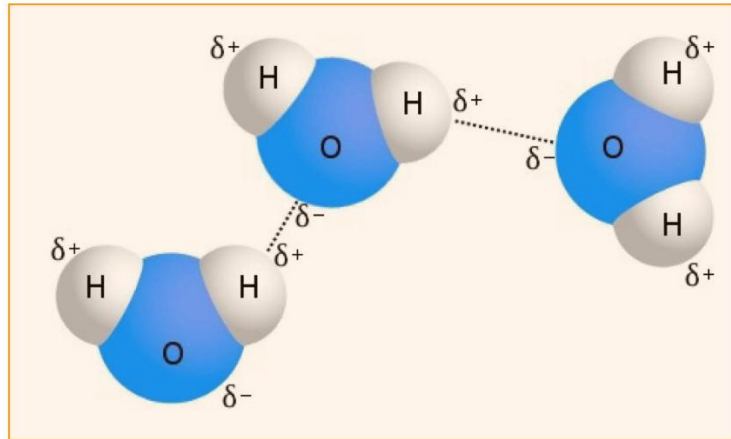


Figure 12: representation of hydrogen bond in water, the partial charges of hydrogen and oxygen atoms are clearly visible.

Electrostatic interactions

Ionic interactions occur through electrostatic attractions between opposite charges or repulsions between similar charges. Amino acids can have charged side chains, with some, like lysine, arginine, and histidine, carrying positive charges, while others, such as aspartate and glutamate, carry negative charges. Additionally, the NH_2 -terminal and COOH -terminal ends of a protein or peptide chain are typically ionized, contributing to overall charge. Charged residues are generally found on the protein surface, where they can effectively interact with the surrounding water, while it is energetically unfavourable for them to be in the hydrophobic core [13].

Protein structure

Secondary structure

Peptide bonds, which are formed between amino acids, force the oxygen, carbon, nitrogen, and hydrogen atoms, along with the adjacent α -carbons, to align in a single plane. Due to this planarity, each residue in the peptide chain has only two degrees of freedom. The peptide chain can rotate around the bond between the α -carbon and the carbon of the peptide bond, as well as around the bond between the nitrogen of the peptide bond and the adjacent α -carbon. Each α -carbon connects two planes formed by peptide bonds. The angle around the $C\alpha-N$ bond is represented by the Greek letter ϕ (phi), while the angle around the $C\alpha-C$ bond is represented by ψ (psi), (fig. 13). In any situation, the complete path of the peptide backbone in a protein can be mapped out if the rotation angles ϕ and ψ are specified. However, certain ϕ and ψ values are prohibited due to steric clashes between nonbonded atoms. An effective way to visualize the range of permissible angles in a protein or a group of proteins is to plot ϕ against ψ values using Ramachandran plot (fig. 14), which shows clusters of ϕ and ψ values in specific regions. Most ϕ and ψ combinations are not allowed because of steric hindrance, resulting in sparsely populated regions on the plot [13].

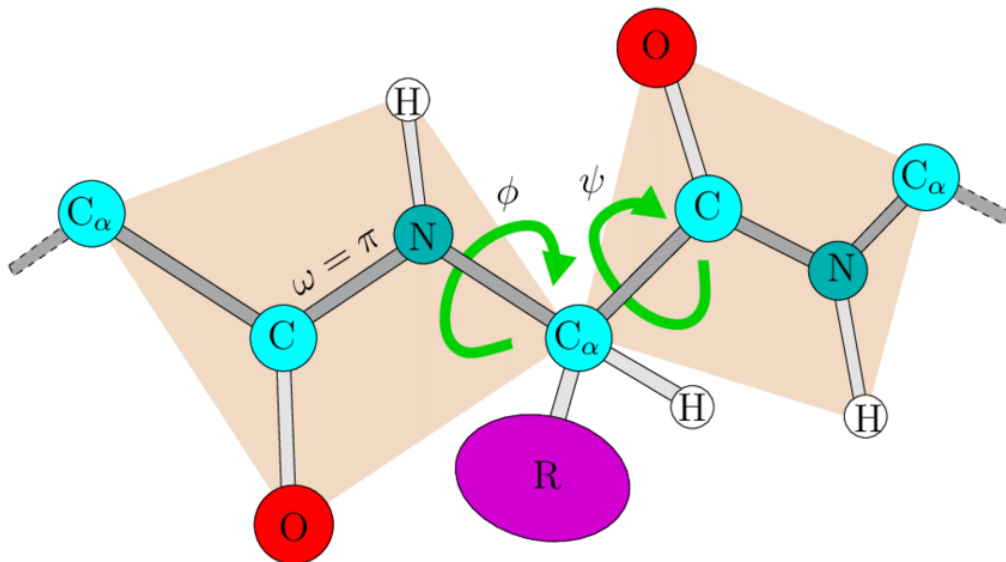


Figure 13. Degrees of freedom of peptide chain: angles ϕ and ψ .

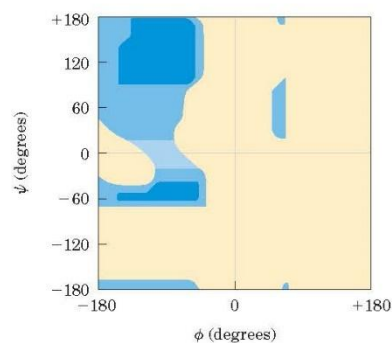


Figure 14. Generic example of Ramachandran plot.

The carbonyl oxygen and amide hydrogen of the peptide bond can form hydrogen bonds with either water molecules in the solvent or with other hydrogen-bonding groups within the peptide chain. These interactions tend to occur cooperatively, involving large segments of the peptide chain. The resulting structures form the protein's secondary structure. When multiple hydrogen bonds form in this manner between parts of the peptide chain, they can give rise to two fundamental structures: α -helices and β -sheets. Segments of the polypeptide chain that do not form α -helices, β -sheets, or turns adopt a conformation known as loops or 'random coils.' These structures are irregular and non-repetitive, typically lacking hydrogen bonds between their amino acids [13].

α -helices

The α -helix is a foundational aspect of protein architecture, capable of organizing up to 35 amino acid residues, whereas the longest β -strands typically encompass around 15 residues. A single α -helix can profoundly influence the stability and overall organization of a protein more than any other individual structural element. In 1951, Pauling identified that the α -helix structure comprises 3.6 amino acid residues per turn, with a hydrogen bond forming between the carbonyl oxygen of residue n and the amide hydrogen of residue $n + 4$. Each hydrogen bond creates a closed loop containing 13 atoms, including the hydrogen atom. So, this structure is known as a 3.6_{13} -helix, denoting the 3.6 residues per turn and the 13 atoms in the hydrogen-bonded loop. Each residue along the helix axis advances by 1.5 \AA [25]. Additionally, the helix moves 5.4 \AA (0.54 nm) along its axis during each complete turn, which is referred to as the translation distance or pitch of the helix. Ignoring the side chains, it has an approximate diameter of 6 \AA . The side chains extend outward from the helix's core, minimizing steric hindrance with the backbone. The hydrogen bonds run parallel to the helix axis, with all carbonyl groups oriented in one direction and all N-H groups pointing in the opposite direction. The full path of the peptide backbone can be determined by specifying the ϕ and ψ angles for each residue. An α -helix forms when ϕ angles are around 60° and ψ angles range from 45° to 50° [13]. As much as 80% of a structure can be helical, and only seven proteins are known that have no helix whatsoever [25].

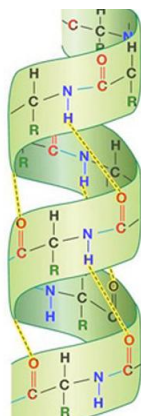


Figure 15. Alpha helix structure.

β -structure: strands and pleated sheets

Another common protein structure formed through local hydrogen bonds is the β -pleated sheet or β -structure, whose discovery is thanks to Pauling and Corey in 1951 and Astbury in 1933 [13], [25]. A β -pleated sheet can be imagined as pleated strips of paper placed side by side, each representing a single peptide strand and called β -strands. A β -pleated sheet can be visualized as zigzagging strips of paper representing peptide strands, with α -carbons at the pleats. β -strands typically consist of 4 to 10 residues and are energetically favorable only when forming β -sheets through hydrogen bonds

between residues on different strands. β -sheets (fig. 16) are formed by the union of two or more such strands [13].

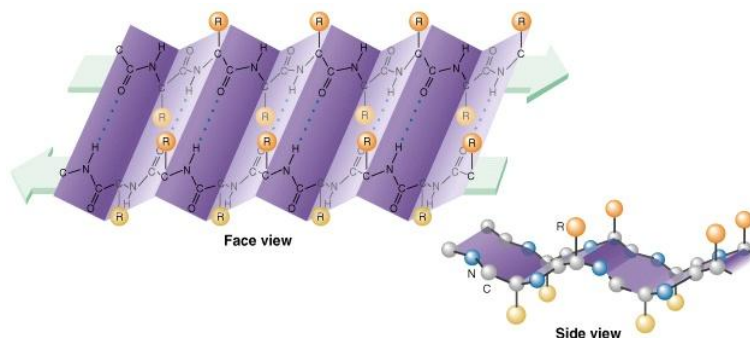


Figure 16. Two beta strands forming a beta sheet.

Furthermore, β -strands can be viewed as special kind of helices with $n = 2$ and a translation of $d = 3.4 \text{ \AA}$ per amino acid (in contrast with the 1.5 \AA of alpha helices). Where p is the pitch per helix and n is the number of residues per turn in the helix. In a β -sheet, the strands are almost fully extended, with ϕ and ψ angles positioned within the broad, shallow energy minimum in the upper left quadrant of the Ramachandran plot (fig. 17).

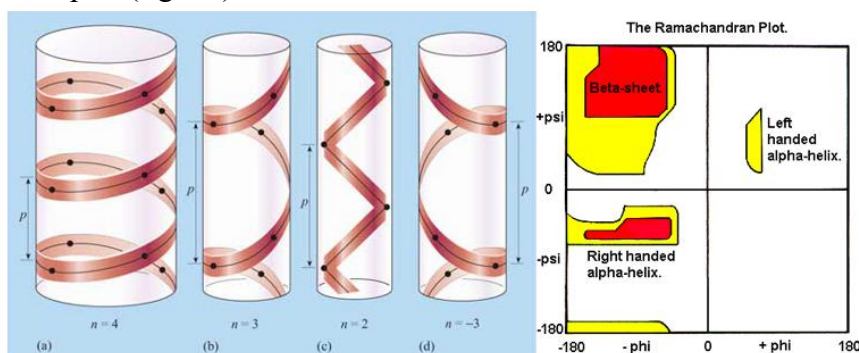


Figure 17. β -strands as particular case of helices and Ramachandran plot of a protein based on its structure

β -strands can form into three types of sheets: purely parallel, purely antiparallel, or mixed, with both parallel and antiparallel pairs. The antiparallel sheet has hydrogen bonds perpendicular to the strands, and narrowly spaced bond pairs alternate with widely spaced pairs while parallel sheet has evenly spaced hydrogen bonds which angle across between the strands. Parallel β -sheets almost always contain at least five strands, while antiparallel β -sheets can consist of just two twisted strands. Parallel β -sheets and the parallel sections of mixed sheets are typically deeply buried within the protein, shielded by other structures such as α -helices. In contrast, antiparallel β -sheets often have one side exposed to the solvent and the other side buried, resulting in an alternation of hydrophobic and hydrophilic side chains along the sequence [25]. Parallel β -structures are less stable than antiparallel ones because they rely on the cooperative effect of an extensive hydrogen-bond network. Antiparallel β -sheets are a key structural component of silk, where the polypeptide chains run parallel to the silk fibers, giving silk its flexibility and limited stretchability. These structures are also found in various other proteins, including immunoglobulin G, superoxide dismutase from bovine erythrocytes, and concanavalin A [13].

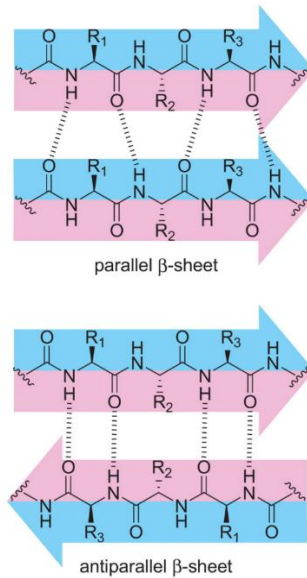


Figure 18. Parallel and antiparallel β -sheet with their own hydrogen bond orientation [26].

One notable characteristic of β -sheets in known protein structures is their twist, resulting from a 30° rotation per residue in a right-handed direction (fig. 19). Parallel β -sheets are less twisted and always buried within the protein, while antiparallel β -sheets can tolerate more distortions and greater solvent exposure. This increased stability of antiparallel sheets, due to their hydrogen bond geometry, aligns with the rarity of small parallel sheets.

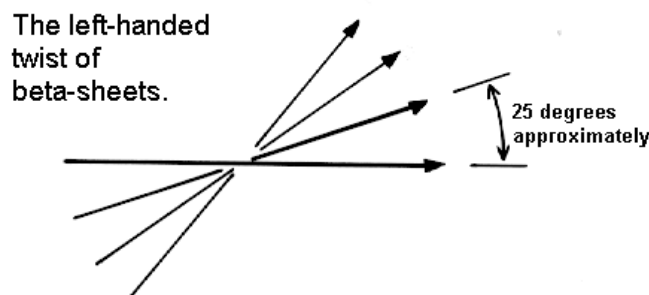


Figure 19. Characteristic twist of β -sheets [25].

Each connection between two β -strands falls into one of two basic categories: hairpin connections, where the backbone chain re-enters the same end of the β -sheet it left, and crossover connections, where the chain loops around to re-enter the sheet at the opposite end (fig. 20). Each connection is named based on how many strands it crosses in the sheet and in which direction, with an "x" added for crossover connections. For example, a "+1" is a hairpin turn, and a "+1x" is a crossover connection between the closest strands; a "+2" is a hairpin turn, and a "+2x" is a crossover connection that skips an intermediate strand in the sheet, and so on [25]. The absolute value of signs is not meaningful since the sheet could be turned upside down.

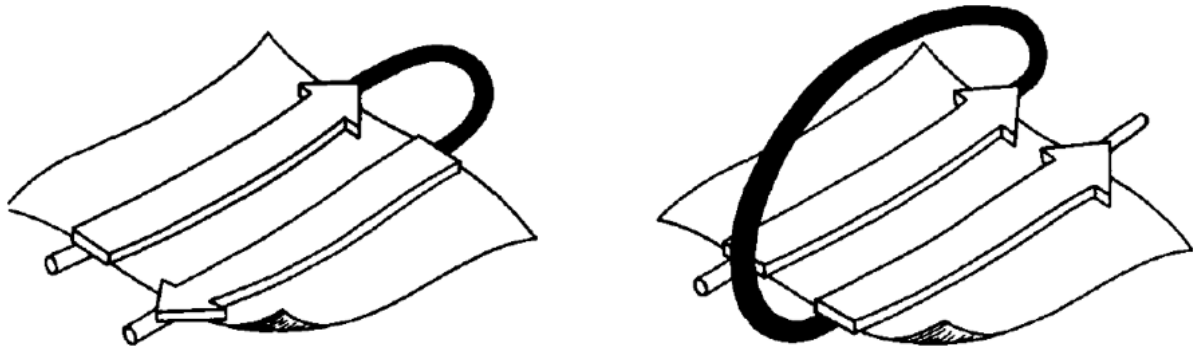


Figure 20. Hairpin (left) and crossover (right) connections [25].

Beta hairpin connection (β -motif)



Figure 21. View of a beta hairpin connection.

The beta hairpin, also called the β -ribbon or β - β unit, is a simple structural motif in proteins (fig. 21). It involves two adjacent β -strands arranged in an antiparallel manner, where the N-terminus of one strand aligns with the C-terminus of the other, connected by a short loop typically containing two to five amino acids. This structure is common in antiparallel beta formations, either as an independent motif or part of more complex beta sheets. This motif does not have a specific functional role [27], [28].

Crossover Connection ($\alpha\beta$ -motif)

A crossover connection involves two parallel β -strands, an α -helix, and two variable-length loops. The α -helix interacts closely with the β -strands, shielding their hydrophobic amino acids from the surrounding solvent. This connection can be visualized as a broad loop of superhelix, starting at one β -strand, passing through the connection, and extending to the second β -strand (fig. 22). The crossover connection can exhibit either a right-handed or left-handed orientation. However, most proteins feature a right-handed crossover connection, which aligns well with the right-handed twist typically observed in β -sheets [28].

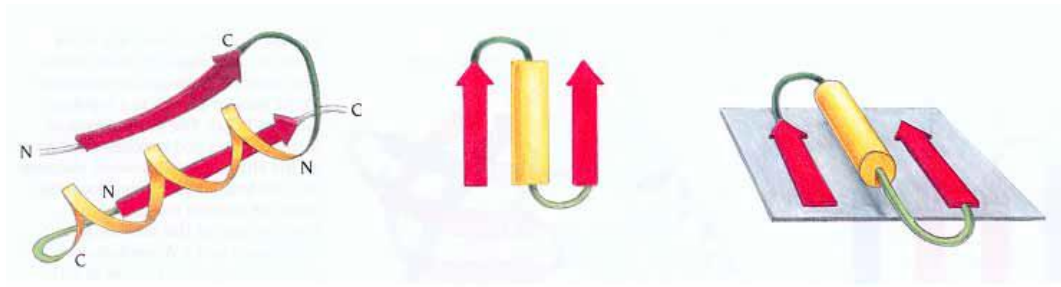


Figure 22: View of the two β -strands and superhelix that form the crossover connection.

Tight turns

Tight turns, also known by various names such as reverse turns, β turns, β bends, hairpin bends, and kinks, are the most common type of non-repetitive protein structure [25]. In this structure, the peptide chain forms a tight loop, stabilized by a hydrogen bond between the carbonyl oxygen of one residue and the amide proton of the residue three positions down the chain. This allows the protein to reverse the direction of its peptide chain. There are two major types of β -turns, but several less common types also exist. Amino acids like proline and glycine frequently appear in β -turn sequences. Glycine, lacking a side chain, is highly adaptable to steric constraints, while proline, with its cyclic structure and fixed ϕ angle, often induces β -turns, aiding in the formation of antiparallel β -pleated sheets [13]. Unlike helices and β structures, which feature consistent ϕ and ψ angles for successive residues, non-repetitive structures exhibit a unique sequence of these angles for each residue, making residue position within the structure more significant. Tight turns typically show a preference for hydrophilic residues, likely because they usually occur at the protein surface where they connect internal secondary structure segments. Tight turns can integrate with other structural elements in various ways. Besides connecting β strands, they often appear at the ends of α -helices. Many protein structures are composed of stretches of secondary structure (helices or β strands) connected by turns, which facilitate changes in direction and offset between sequential secondary structure segments. Tight turns are particularly effective in forming α - α and α - β joints, but their most common role is in hairpin connections between adjacent antiparallel β strands, where the turn's hydrogen bond also contributes to the β sheet's hydrogen bonding network. Due to strong residue preferences within tight turns, they likely influence the folding of adjacent strands [25].

β -bulge

The β bulge (Richardson et al., 1978) is a minor, nonrepetitive structure that can exist independently within coil regions but is most seen as an irregularity in an antiparallel β structure. A β bulge is characterized as an area between two consecutive β -type hydrogen bonds, comprising two residues on one strand opposite a single residue on the other strand. Only about 5% of β bulges occur between parallel strands, with the majority in antiparallel strands situated between closely spaced hydrogen bonds. Bulges, along with tight turns, frequently appear at active sites due to their precise and localized impact on side chain orientation. Another potential role for β bulges is to accommodate single-residue insertion or deletion mutations without completely disrupting the β sheet structure. Furthermore, β bulges alter the direction of the backbone strands, enhancing the right-handed twist of the strands. This makes them valuable in shaping large features of β sheets and extended hairpin loops. For example, in antiparallel β barrels, a strong local twist is essential for closing barrels composed of five or six strands.

Disulfide bridges

Disulfide bridges are true covalent bonds formed between the sulfur atoms of two cysteine side chains and are generally considered part of the primary structure of a protein. However, proteins usually fold correctly even without disulfide formation, suggesting that these SS links affect the structure more like secondary structural elements by providing local specificity and stabilization.

Random coil

Polypeptide chain segments that are not α -helices, β -sheets, or turns take on a conformation called loops or 'random coils,' which are non-repetitive, irregular structures often lacking hydrogen bonds between the constituent amino acids [25]. The segments of the polypeptide chain that do not form any type of secondary structure are considered irregular, with amino acid (ϕ , ψ) pairs taking on various and non-constant values. These regions are generally flexible and capable of adopting multiple conformations, making them challenging to detect experimentally. For instance, N-terminal and C-terminal ends and areas rich in charged amino acids (particularly lysine) often display such behavior. Besides linking secondary structure elements, these regions can also contribute to the formation of binding sites and active sites in enzymes, remaining disordered in the absence of a specific molecule and becoming ordered upon binding [28].

β -barrels

Large antiparallel β -sheets often fold partially or completely into cylindrical shapes called β -barrels. These barrels can contain between 5 and 13 strands, with interiors packed with hydrophobic side chains. Despite variations in strand number, the cross sections of all β -barrels are quite similar, showing slight flattening in one direction. Barrels in protein structures look uniform because their twist can be adjusted. Similar to how bias-woven bandages tighten when stretched, barrels with more strands have a smaller diameter when the twist is less.

The initial category of antiparallel β barrels features straightforward up-and-down +1, +1, +1 connections throughout (fig. 23). While it is rare for a barrel to consist solely of up-and-down strands, many larger barrels and sheets incorporate sections with four to six strands that follow this simple topology.

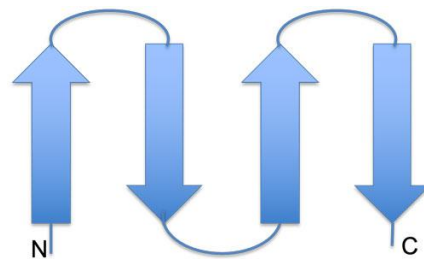


Figure 23. "up and down" motif.

However, the most common type of antiparallel β barrel structures features a Greek key topology (fig. 24).

The Greek key motif consists of four β -strands, characterized by -3, +1, +1 connections or slight variations of this pattern. More in depth, three of these strands follow an "up and down" topology and are connected by β -hairpins. Then, a longer connection links to the fourth β -strand, which is adjacent to the first [28], [25].

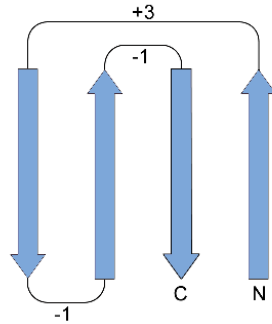


Figure 24. “Greek key” motif.

Tertiary structure

The folding of a single polypeptide chain into a three-dimensional shape is known as its tertiary structure. All the information required to fold the protein into its native tertiary structure is contained within its primary structure. Through extensive research, some key principles were discovered. Firstly, secondary structures—helices and sheets—form whenever possible due to numerous hydrogen bonds. Secondly, α -helices and β -sheets often pack closely together within the protein, as single-layer structures are not stable. There are a few common methods for this packing to occur. Thirdly, the peptide segments between secondary structures tend to be short and direct, preventing complex twists and knots as the peptide moves between regions of secondary structure. As a result, protein chains usually fold so that secondary structures are arranged in a few common patterns, leading to families of proteins with similar tertiary structures despite little evolutionary or functional relationship. Ultimately, proteins fold to form the most stable structures possible, with stability arising from the formation of numerous intramolecular hydrogen bonds and the reduction in surface area exposed to solvent upon folding.

Structure based protein classification: fibrous and globular

Essentially proteins can be divided into two categories depending on their conformation in space and consequently function.

Fibrous proteins are made up of polypeptide chains that are elongated and fibrous in nature or have a sheet like structure. These fibers and sheets are mechanically strong and are water insoluble. They are often structural proteins that provide strength and protection to cells and tissue. The α -keratins are fibrous proteins involved in the structure of hair, fingernails and horns, and their secondary structure is the α -helix with a higher level of structure being the coiled coil. Fibroins (a β -keratin) are fibrous proteins making up silk and spider webs, and their secondary structure is β -sheets. Collagen is an abundant fibrous protein in vertebrate animals being found in tendons, cartilage and bone, and it has a unique structure. Elastin is an important component of tissues, such as ligaments and skin, and is highly elastic [29].

Globular proteins, named for their roughly spherical shape, are the most prevalent proteins found in nature. These proteins exhibit a vast array of three-dimensional structures. Almost all globular proteins include a significant number of α -helices and β -sheets, which fold into a compact form stabilized by interactions between both polar and nonpolar groups. The three-dimensional structure of globular proteins forms spontaneously and is maintained through interactions among the amino acid side chains. Typically, the hydrophobic side chains are tucked away inside the protein, densely packed to avoid contact with water, while the hydrophilic side chains are positioned on the surface, exposed to the aqueous environment. This arrangement makes globular proteins highly soluble in

water. The structural diversity of globular proteins underpins their wide range of functions, including binding, catalysis, regulation, transport, immunity, and cellular signaling, among others [30].

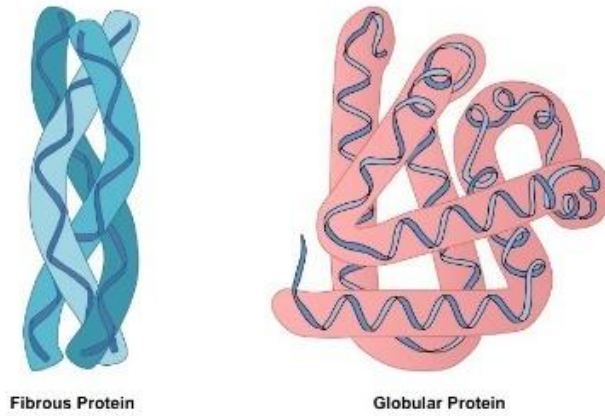


Figure 25. Cartoon representing the two types of protein, fibrous and globular [31].

SOD protein family

Function and importance

Our cells depend on oxygen to survive as the final electron acceptor in respiration, which allows us to derive much more energy from food than would be possible otherwise. However, oxygen also poses dangers. Reactive oxygen species, such as superoxide (an oxygen molecule with an extra electron), can cause cellular damage. This superoxide as many others reactive oxygen species (ROS) may induce DNA mutations or damage enzymes responsible for producing amino acids and other crucial molecules. The result is oxidative stress which is linked to many human diseases including hypertension, diabetes mellitus, ischemic diseases, and malignancies. To fight this potential danger, most cells make some globular proteins called superoxide dismutase (SOD), metalloenzymes that convert superoxide into a less toxic substance through the dismutation reaction. All organisms have developed different types of SOD enzymes (fig. 26) that utilize various metal ions for electron-shuffling reactions. In humans, three types of SOD are present: copper-zinc SOD inside the cytoplasm of cells (even though a small portion is found in the intermembrane space of the mitochondria), another form with an adhesive tail for external structures, and manganese SOD in mitochondria. Bacteria produce diverse SOD enzymes, including iron and nickel variants. The copper-zinc superoxide dismutase protein, also known as SOD1, is of particular interest because mutations in this protein are found in patients with familial ALS and it will be the protein on which the work is focused on.

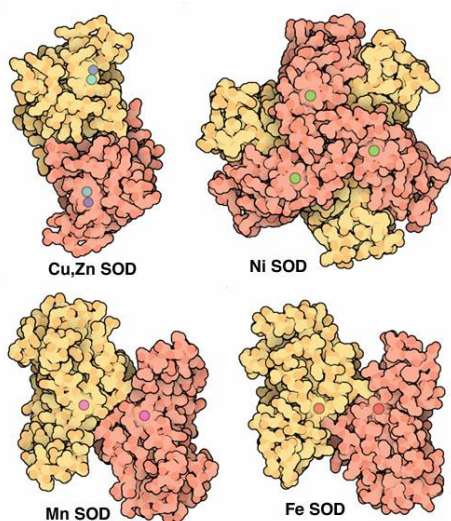


Figure 26. Different types of SOD proteins in human and bacteria.

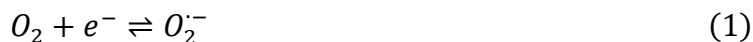
Dismutation reaction

Dismutation is a process where two identical but opposing reactions (a reduction and an oxidation) happen on separate molecules [32]. Atoms of the same element undergo reduction and oxidation at the same time, transitioning from a single oxidation state to two distinct oxidation states. Essentially, a compound with an intermediate oxidation state is converted into two compounds, one with a higher and one with a lower oxidation state. This means a single species is both reduced and oxidized, resulting in the formation of two different products [33]. The substrate of the dismutation process in SOD proteins is superoxide anion, whose Lewis structure is shown in figure 27. This chemical species O_2^- contains both a negative electrical charge and an unpaired electron, making it both an anion and a radical at the same time (radical ion).

Furthermore, since the negative charge and the unpaired electron are located, one on each oxygen atom, it also represents a radical ion. The presence of the unpaired electron makes this molecular ion

paramagnetic [34]. Superoxide is generated in aerobic organisms as a byproduct of metabolism. The superoxide anion can serve as an intermediate or a byproduct in metabolic processes like respiration or photosynthesis [35].

The monovalent reduction that gives $O_2^{\cdot-}$ from O_2 is the following one [36]:



Superoxide ion ($O_2^{\cdot-}$) exists in equilibrium with its protonated form HO_2^{\cdot} (hydroperoxyl radical) [37]:



The enzyme takes two molecules of protonated superoxide, removes an extra electron from one, and transfers it to the other. As a result, one molecule loses an electron, forming regular oxygen, while the other gains an additional electron and quickly combines with two hydrogen ions to produce hydrogen peroxide H_2O_2 [38].

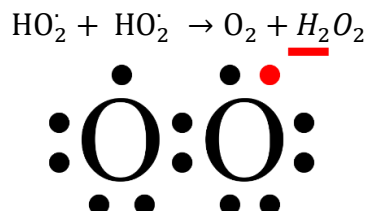


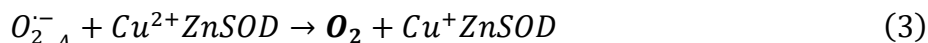
Figure 27: Lewis structure of superoxide ion with highlighted in red the unpaired electron and the negative electrical charge.

SOD1

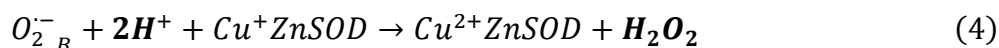
SOD1 dismutation reaction

It was explained the general mechanism that regulates dismutation reaction in SOD class proteins. However now, we take a look in detail at the mode of action of the Cu,Zn superoxide dismutase, which is referred to as SOD1.

The reaction begins with the binding of superoxide radical to the active site of SOD1 that contains a copper ion. Cu^{2+} is reduced to Cu^+ while the superoxide molecule, $O_2^{\cdot-}$ (which donated an electron) is oxidized to O_2 .



Another superoxide ion binds to the reduced copper ion in the active site. So, in this step Cu^+ is oxidized to Cu^{2+} , while the superoxide molecule acquires the electron from Cu^{2+} and together with two protons ($2H^+$) it's reduced to hydrogen peroxide, H_2O_2 [35].



The overall redox is:



A simplified catalytic cycle can be seen in the figure 28 [35].

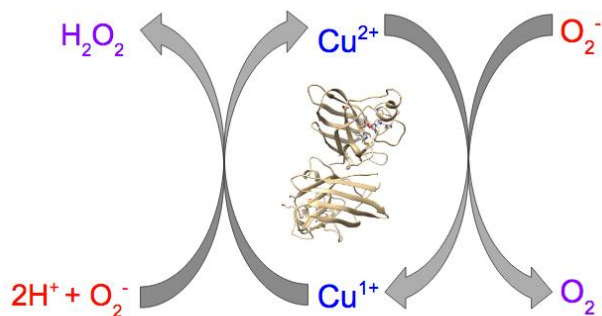


Figure 28. Catalytic cycle of SOD1 dismutation [35].

Cu/Zn superoxide dismutase is an exceptionally effective enzyme. It was found that one in every ten collisions between superoxide and the enzyme results in a reaction. This efficiency is surprising because the active site occupies only a tiny part of the enzyme's surface, suggesting that most collisions might occur elsewhere on the enzyme. However, the shape and properties of the active site provides clues to this efficiency and there are depicted in figure 29 [38].

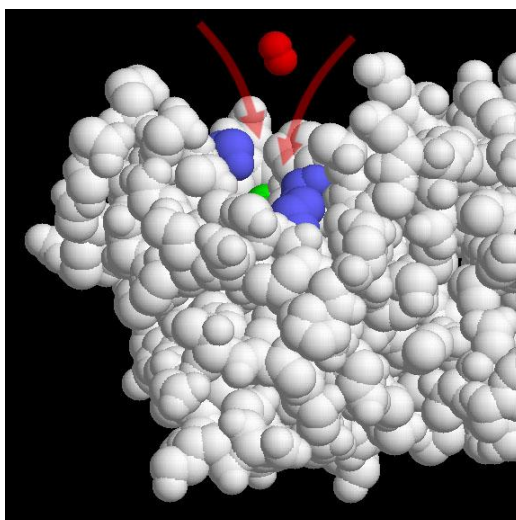


Figure 29. The active site is funnel-shaped, with copper and zinc ions (colored green) located at the base. The strong positive charge of these metal ions, coupled with nearby positively charged amino acids (colored blue), attract the negatively-charged superoxide molecules (colored red) into the funnel [38].

Hydrogen peroxide is also hazardous, so the cell must utilize the enzyme catalase to neutralize it [32], with following the subsequent reaction:



Catalases are exceptionally fast enzymes capable of breaking down millions of hydrogen peroxide molecules per second. Some examples are shown in figure 30 [38].

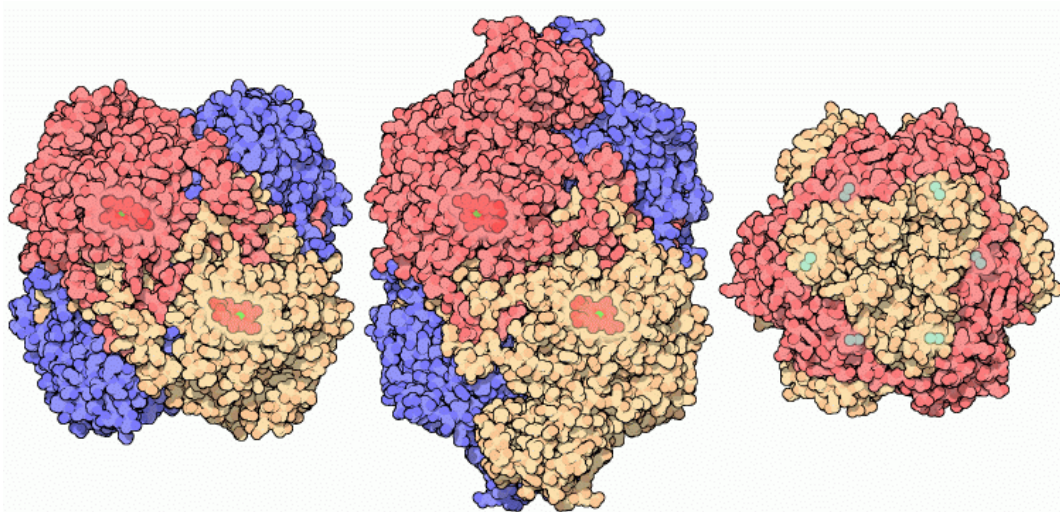


Figure 30. The catalase that protects human red blood cells, shown in PDB entry 1qqw, consists of four identical subunits and uses a heme iron to catalyze the reaction. Other bacteria, as shown in PDB entry 1iph, eliminate hydrogen peroxide using a larger catalase enzyme with a similar structure around the heme iron. Yet other bacteria use a completely different catalase that employs manganese ions instead of heme iron, as depicted in the figure on the far right (PDB entry 1jku).

SOD1 structure

The SOD1 protein is a homodimer, meaning it consists of two identical subunits, shown in contrasted colors in figure 31 A, related by a two-fold symmetry axis. The subunits are positioned such that their active sites are oriented in opposite directions. Each subunit is made up of 153 amino acid residues and consists of a β barrel (fig. 31 B) composed of eight β -strands arranged in a Greek key motif (fig. 32 A) that form two antiparallel β -sheets (fig 32 B). The core of the barrel is formed by tightly packed hydrophobic residues and loops β 3/ β 4 (III) and β 6/ β 7 (VI) form the +3 β -strand Greek key connections. At the active site of each subunit, each one formed by two alpha helix featured external loops [39], there is one copper ion and one zinc ion. The first loop, β 4/ β 5 (IV) or metal zinc binding loop, is forms one of the -1 β -strand Greek key connections (the other one is formed by loop V) but most importantly links the dimer interface with active site zinc and contains a stabilizing disulfide bond. The second, β 7/ β 8 (VII) or electrostatic loop, which contains the copper ion, guides and accelerates the substrate O_2^- into the active site. During the catalysis, the copper ion functions as the center for electron transfer, whereas the zinc ion provides structural support acting as a scaffold and contributes to the positive charge of the active site, helping to attract the negatively charged superoxide anion [35], [40]. The Cu and Zn sites are located just outside the β -barrel within the active site channel, serving both structural and catalytic functions. Hydrophobic anchors secure the β -barrel to the Cu ion in the active site. Figure 33 shows how metal ions bind to protein residues. The active site is selectively permeable, allowing the entry of superoxide while excluding larger anions like phosphate [40].

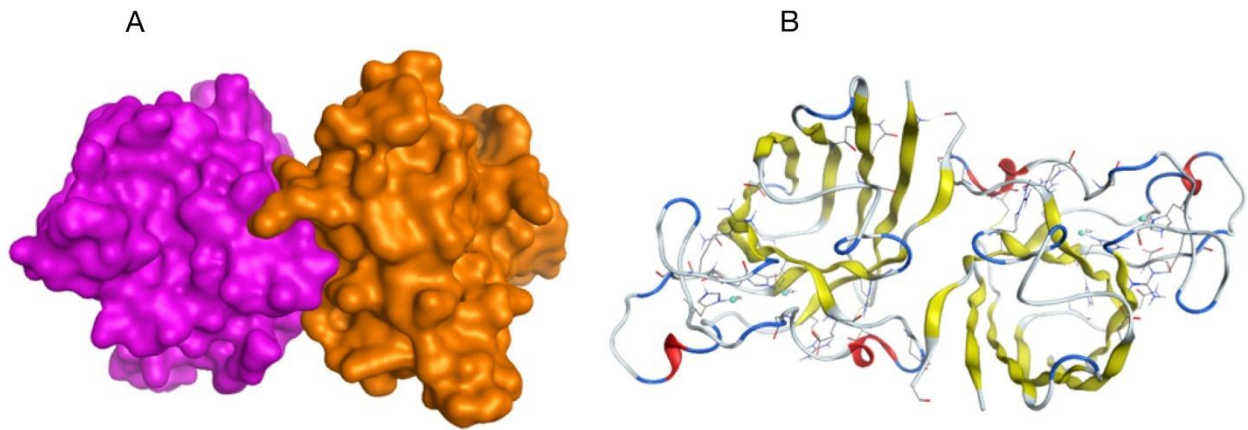


Figure 31. 3D structure of SOD1 dimer from pdb. **A:** SOD1, being a homodimer, has two identical monomers, here shown in purple and orange. **B:** According to the RasMol structure color scheme, the beta strands that make up the beta sheets, which in turn form the beta barrels (visible as if they were almost in cross-section), are distinguished in yellow; the random coil structures are shown in white; the alpha helices are in red; the turns are in blue.

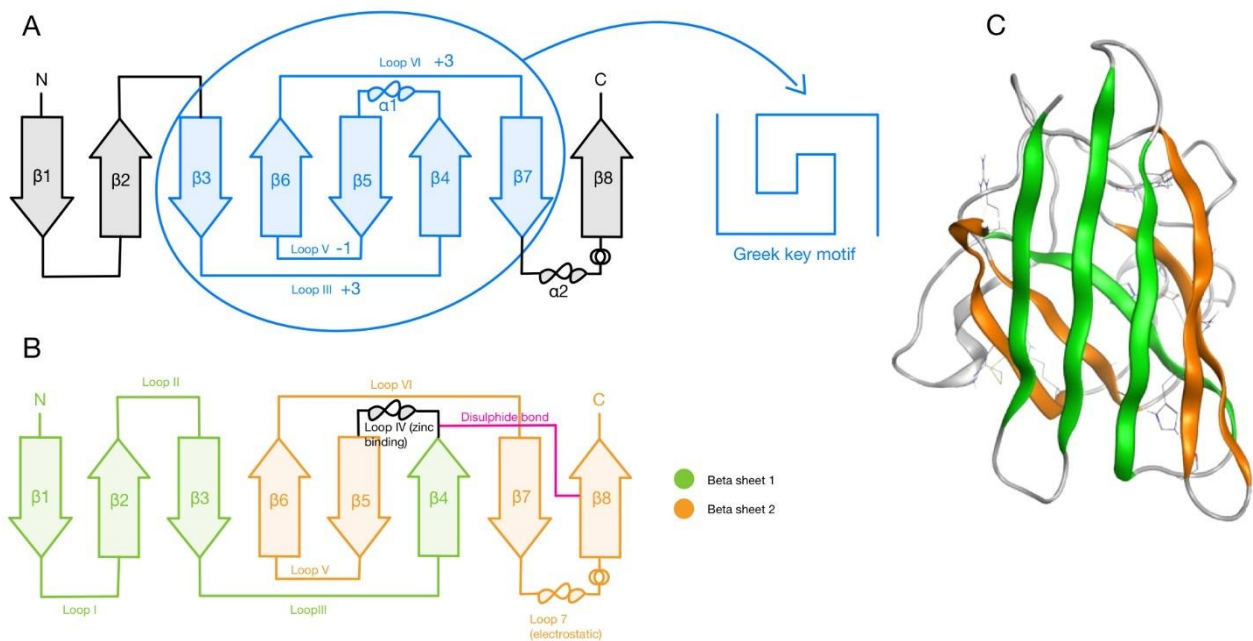


Figure 32. Schematic structure of SOD1 monomer. **A:** It's depicted the secondary structure of the protein with highlighted by a blue line the Greek Key motif made of four strands ($\beta 3$ - $\beta 7$) and four loops (III-VI), of which two show an alpha helix structure. **B:** The image is analogous to A, but the beta strands belonging to the first and second beta sheets are distinguished in green and orange. All the loop names are shown, including the main ones relevant to the protein's activity, the zinc-binding loop and the electrostatic loop. The disulphide bond, necessary for improved protein stability is shown too in purple. **C:** The 3D structure of the monomer can be visualized, with the strands of the corresponding beta sheets highlighted using the same consistent coloring as in B.

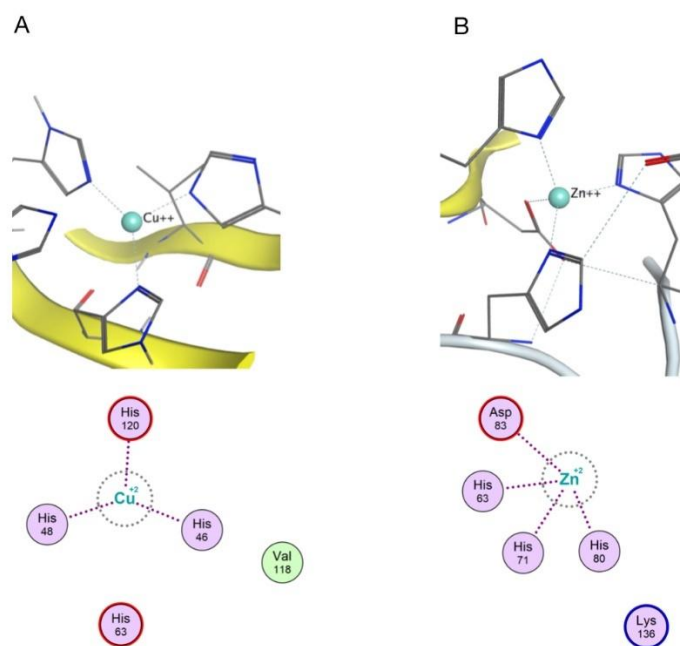


Figure 33. **A:** the Cu ion is bonded to three histidines (46, 48 and 120). **B:** the Zn ion bonded to one aspartic acid and three histidines (83, 63, 71, 80). In each case all side chains extending beyond the β -barrel.

Description of SOD1 dismutation reaction based on structure

In the initial phase, the superoxide anion (O_2^-) is stabilized by Arg143 through electrostatic interactions. An electron from the oxygen radical is transferred to Cu^{2+} , reducing it to the less stable Cu^+ state. This reduction breaks the bridge between copper and His63, allowing the eta-nitrogen of His63 to become protonated. The newly formed oxygen molecule is then released from Arg143. In the subsequent phase, Cu^+ donates an electron to O_2^- . The superoxide anion can now form two covalent bonds by acquiring two hydrogen atoms: one from the protonated His63 and the other from H_3O^+ present in the active site. This reaction produces hydrogen peroxide (H_2O_2). With copper returning to its Cu^{2+} oxidation state, the bridge between His63 and copper is re-established. Another key residue in the active site is Asp124, located in the electrostatic loop. Asp124 forms hydrogen bonds with His46 and His71, creating a secondary indirect bridge between copper and zinc. This bridge is thought to enhance catalytic efficiency [41].

SOD1 folding and maturation

To perform biological roles such as catalysis, transport, and signaling, proteins need to establish and preserve specific 3-dimensional shapes with the right dynamic characteristics. For SOD1, the folding and maturation start when the ribosome synthesizes the nascent SOD1 chain. After being released from the ribosome, this disordered chain undergoes a cooperative folding process, transforming directly into the folded apoSOD1 monomer. This type of folding, known as two-state folding, is common among globular proteins. The folded apoSOD1 monomer undergoes further maturation and stabilization through metal binding, the formation of disulphide bonds, and dimerization. The primary contributor to the stability of the folded SOD1 monomer is metalation, where Zn^{2+} and Cu^+/Cu^{2+} play distinct roles. Zn^{2+} has a high affinity for both Cu and Zn binding sites, facilitating a Zn^{2+} mediated folding process. Initially, Zn^{2+} transiently binds to the Cu ligands within the folding

nucleus, then moves to the higher-affinity Zn^{2+} site, establishing the native state. In vivo, the reactive Cu^+/Cu^{2+} ion is delivered to the SOD1 monomer by the copper chaperone CCS. In addition to metalation, although monomeric SOD1 can fold without forming the native intra-subunit C57-C146 disulfide bond, this bond significantly enhances the monomer's stability by anchoring loop IV to the β -barrel [42], [43]. For proper functionality, SOD1 also needs the exposure of a hydrophobic region necessary for the organization of SOD1 monomers into dimers thanks to hydrogen bonding interactions [44]. When the disulfide bond is reduced and both metal ions are removed, loop IV will "swing" outward, leading to the dissociation of the dimer [43].

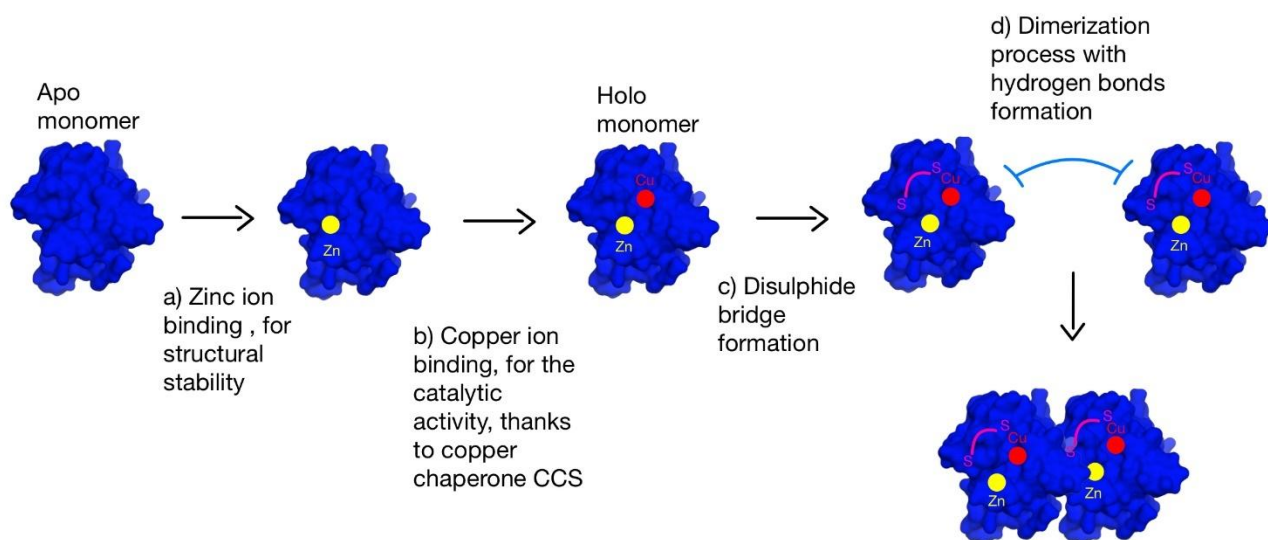


Figure 34. Illustration of the multistep folding and maturation process of SOD1 holo dimer.

SOD1 mutations in ALS

Mutations in superoxide dismutase 1 gene (SOD1) are linked to ALS [45].

Up to now, nearly 200 point mutations in the SOD1 gene, and consequently in the SOD1 protein, have been identified [46]. The list with each amino acid modification can be seen in the alsod database [47]. In this work, seven point mutations were considered, divided into two groups based on the position of the modified amino acid. The first group consists of mutations located in the region where the active site is present, occurring within the protein near the metal ion binding site, and includes the G85R mutation [48]. The second group includes the A4V, G41D, G41S, and G93A mutations, which, despite involving regions distant from the active site, could destabilize the protein in terms of dimer stability or weaken the disulfide bonds that help maintain the stability of the dimer interface [49].

From loss of function to toxic acquired behaviour

Given SOD1's function as an antioxidant enzyme, it was initially believed that the mutations in SOD1 linked to familial ALS (fALS) caused a loss of enzymatic activity. This loss was thought to result in cellular toxicity due to elevated levels of oxygen radicals. However later studies revealed that many SOD1 mutants possess a high enzymatic activity and that SOD1 is not indispensable to cell survival [50], being its antioxidant function compensated by some other members of SOD protein family [51]. After dismissing the idea that SOD1-related familial ALS was due to a loss of function, researchers shifted their focus to exploring mechanisms involving a gain of toxic function.

Mutations-induced toxicity can occur through two mechanisms. The first is related to improper metal binding: if the mutated enzyme can no longer bind zinc ions, which are crucial for maintaining the

active site, it will also be unable to bind the copper necessary for the dismutation reaction. As a result, the enzymatic activity may reverse, leading to excessive production of superoxide ions [52]. Additionally, reduced metal binding can cause the release of metal ions, which poses a risk of metal-induced neurotoxicity [53]. The second mechanism, discussed in the following chapter, is related to the ability of mutations to induce the protein to lose its capacity to fold correctly, causing misfolding and, in some cases, subsequent protein aggregation.

Prion like mechanism of conversion of SOD1

The native SOD1 protein exhibits remarkable thermodynamic and kinetic stability, thanks to the zinc ions acting as a scaffold, taking together the active site and to the internal disulfide bond [54]. Despite the native SOD1 enzyme's remarkable stability, certain mutations can destabilize it, leading to misfolding and aggregation [55]. Introducing preformed SOD1 aggregates in vitro has been shown to speed up the aggregation of soluble SOD1, indicating that misfolded SOD1 can promote further misfolding of the protein [56]. Misfolded proteins can function as seeds in a nucleated polymerization process, converting native proteins into pathological forms and leading to fibril formation [57]. SOD1 aggregates can enter cells and initiating the aggregation of the normally soluble protein within them. This induced aggregation can continue and spread from one cell to another [58]. Recent findings suggest that many ALS-related mutations share a common issue: they increase the tendency of the protein to expose hydrophobic surfaces that are usually hidden inside the native structure [59]. The location where the disease first appears might be where protein misfolding initially starts. Once this misfolding begins, the malformed protein may propagate from the original site to neighboring areas, acting like a prion. A prion is a type of infectious protein characterized by a distinct structural conformation compared to its normal counterpart. This altered form can induce similar conformational changes in other molecules of the same protein, allowing the abnormal structure to propagate [60]. It spreads by converting normal proteins into its abnormal form without needing other proteins, factors, or genetic material. Just like a prion, it's believed that the misfolded SOD1 proteins can transmit their structural and space configuration information to wild type ones.

Two modes of action have been proposed to explain this process. The first, known as template-directed misfolding is a process where a misfolded protein induces the misfolding of a native protein (fig. 35). In the case of SOD1, this process primarily involves monomers, with a misfolded monomer acting as a "template" for the misfolding of a wild type monomer [61], [62]. Below are the steps of the mechanism, focusing on the interaction between misfolded and wild type monomers.

- **Initial Trigger:** The process begins with the formation of an SOD1 monomer in a misfolded conformation. This can occur due to oxidative stress, lack of binding with metal ions (such as copper or zinc) and of course genetic mutations as in our case.
- **Exposure of Hydrophobic Surfaces:** Misfolding exposes hydrophobic surfaces that are normally hidden within the native protein structure. These exposed surfaces are crucial for the template-directed misfolding process.
- **Recognition and Binding:** The misfolded monomer interacts with a wild type SOD1 monomer. This interaction is mediated by the exposed hydrophobic surfaces or specific protein domains. The misfolded monomer acts as a "template" or model for the wild type monomer.
- **Induction of Misfolding:** The contact between the misfolded monomer and the wild type one induces a conformational change in the wild type monomer. This change drives the wild type protein to lose its native structure and adopt a conformation similar to that of the misfolded one.

- Dissociation of the Dimer: The resulting dimer, in which at least one monomer is misfolded, may be unstable. This instability can lead to the dissociation of the dimer into two separate monomers.
- Propagation of Misfolding: Once separated, the misfolded monomers can seek out other wild type monomers to interact with. When they bind to a new wild type monomer, they induce it to misfold, thus continuing the propagation cycle.

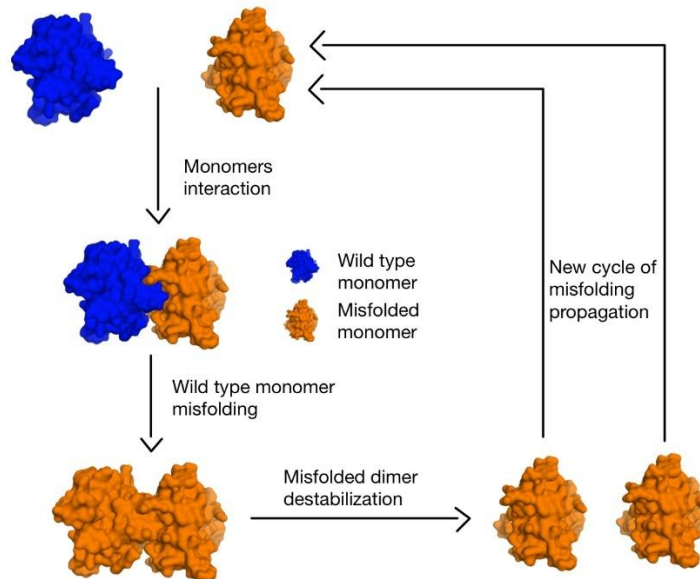


Figure 35. Step by step illustration of template-directed misfolding process.

The second mechanism is called seeded polymerization, where a misfolded protein (or a small aggregate of misfolded proteins) acts as a "seed" for the growth of amyloid fibrils (fig. 36). This process is crucial in various neurodegenerative diseases, including those related to the SOD1 protein [63]. Below, I describe the various steps, focusing on the role of wild type and misfolded monomers.

- Initiation of the Process: The process begins with the formation of a nucleus or "seed" of misfolded proteins. This seed can be an oligomer of a few misfolded SOD1 monomers or a small fibril that has formed spontaneously or through an initial misfolding event.
- Critical State: The formation of this seed is generally a slow step and requires a minimum number of misfolded monomers to overcome an energy barrier and form a stable structure that can serve as a starting point for polymerization.
- Interaction with Wild Type Monomers: Once the seed is formed, it begins to interact with wild type SOD1 monomers. The wild type monomers approach the seed and, through interaction with the seed's surface, undergo a conformational change.
- Conversion of Wild Type Monomers: During this interaction, wild type monomers are converted into a misfolded form similar to that of the monomers present in the seed. This conformational change allows wild type monomers to integrate into the seed structure, elongating the fibril.
- Continued Addition of Monomers: The process of adding converted wild type monomers continues, growing the fibril. Each new monomer that attaches to the fibril further stabilizes the structure and facilitates the addition of more monomers.
- Formation of Amyloid Fibrils: The continued incorporation of misfolded monomers leads to the formation of a long, stable amyloid fibril. These fibrils are characterized by a highly ordered beta-sheet structure, typical of amyloid fibril.

- **Fibril Fragmentation:** As fibrils grow, they may fragment, creating new seeds. These fragments, being already in a misfolded state, can act as new nuclei for polymerization, accelerating the propagation of misfolding.
- **Cellular Propagation:** Fibrils or their fragments can be released from cells or transported to other regions, spreading the process to other areas of tissue or to other cells.
- **Formation of Inclusions:** Over time, fibrils can further aggregate, forming large deposits or intracellular inclusions. These aggregates are often insoluble and can interfere with cellular functions, contributing to neurodegeneration.
- **Pathological Impact:** These fibrillar aggregates are associated with cellular toxicity, as they can disrupt various cellular processes such as intracellular trafficking, mitochondrial function, and protein turnover, contributing to disease progression.

The template directed misfolding involves a misfolded protein being more stable than its native form, but this stability is only achieved through catalytic interaction with another misfolded protein. The second mechanism, called nucleation or seed polymerization, involves the misfolded monomer being initially less stable than the native monomer but gaining stability through aggregation into a multimeric form giving born to fibrils which may have strain-like properties who can then propagate across the cell barrier from cell to cell [64].

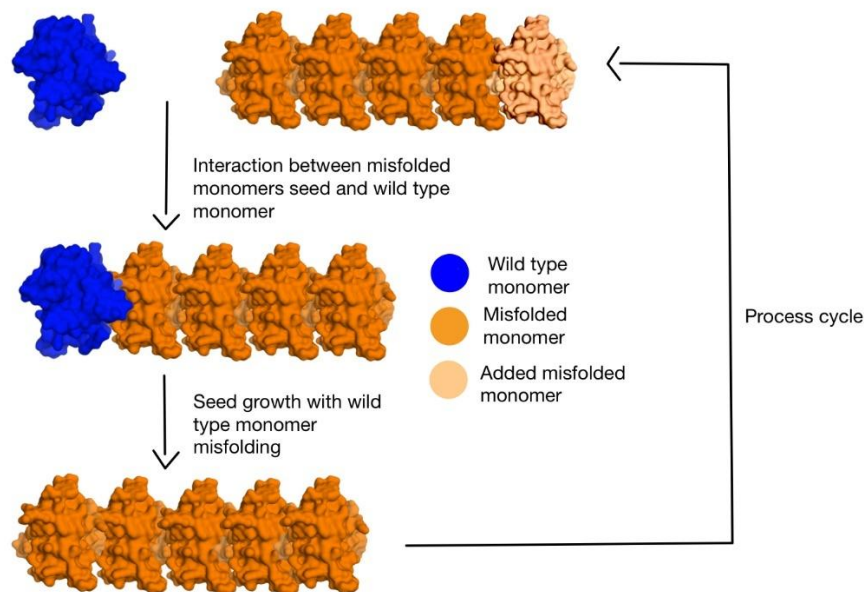


Figure 36. Step by step illustration of seeded polymerization process. At the conclusion of each cycle, the seed is increased by one monomer, which leads to the formation of fibrils and/or aggregates.

Aim of the work

Given that SOD1 is one of the first proteins discovered to be implicated in familial ALS (FALS), with malfunction likely caused by numerous mutations identified over the years and now collected in the ALSOD database, my work initially focused on understanding whether and how the presence of one of these selected mutations, applied to one of the monomers in the SOD1 dimer, could affect its structure and spatial conformation compared to that of a wild-type dimer. Secondly, computational research was conducted to identify potential inhibitor drugs that could prevent prion-like infection mechanisms based on template-directed misfolding by stabilizing the dimer interface, thus preventing dimer destabilization and blocking the propagation of the disease.

Proposed mechanism for prion like conversion inhibition

The approach of placing a drug at the interface between a misfolded SOD1 monomer and a wild type monomer to stabilize the dimer and prevent the misfolding of the wild type monomer is a strategy that could potentially disrupt the pathological cycle of misfolding and aggregation associated with the prion-like conversion of SOD1. Let's explore how this could work in the two mechanisms:

1. Template-Directed Misfolding

- In the context of template-directed misfolding, the drug would aim to prevent the process in which the misfolded monomer acts as a template to induce the misfolding of the wild type monomer (fig. 37). If the drug can stabilize the interface between the two monomers, it would prevent the critical interaction that leads to the misfolding of the wild type monomer. This approach could be particularly effective because:
 - **Blocking the Induction of Misfolding:** If the interface is stabilized and the wild type monomer retains its native conformation, the templating effect of the misfolded monomer would be neutralized.
 - **Disruption of the Propagation Cycle:** This could interrupt the propagation of misfolding by interfering with dimer destabilization, reducing the formation of new misfolded aggregates.

2. Seeded Polymerization

- In the context of seeded polymerization, the drug could influence two aspects:
 - **Prevention of Aggregate Growth:** By stabilizing the interface, the drug might prevent further addition of wild type monomers to the aggregate, thereby inhibiting seed growth and amyloid fibril formation.
 - **Stabilization of Dimers:** If the drug can stabilize the dimer and prevent dissociation into monomers, it could prevent wild type monomers from dissociating and joining an existing seed, thus reducing the likelihood of polymerization and aggregation.

The drug strategy is particularly well-suited for the template-directed misfolding mechanism, as it aims to prevent the critical interaction that induces misfolding, thereby effectively stopping the misfolding cycle and halting the propagation of the pathological process. In the case of seeded polymerization, the drug could also be beneficial, but its effectiveness would depend on the point of application; it could prevent seed growth or dimer dissociation but might not entirely stop the initial nucleation or the processes that precede seed formation.

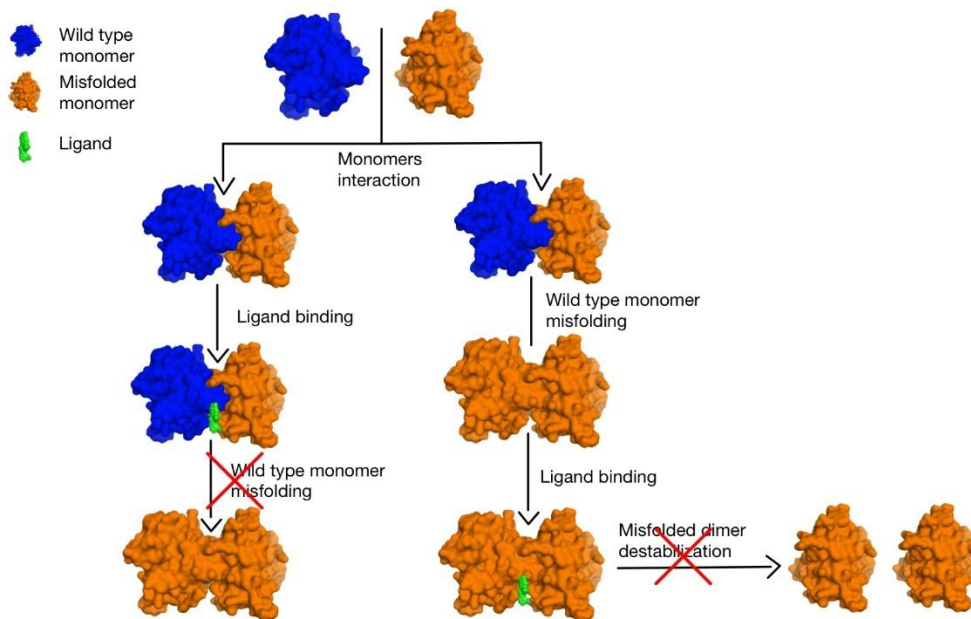


Figure 37: Possible mechanism of misfolding inhibition by a ligand, based on the template-directed misfolding.

Methods

Obtaining the structure of the human wild-type holo protein

From the RCSB PDB database, the pdb file of the human SOD1 protein structure was downloaded under the code '3ecu'. It includes two SOD1 dimers, each consisting of identical monomers, in accordance with the theory. However, this structure does not include the metal ions (it's an apo structure), which, as previously mentioned in the initial overview, are fundamental in ensuring dimer stability during its enzymatic activity. Therefore, in order to integrate these components into the structure, the following steps were followed with the aid of MOE software:

- A structure of animal SOD1, with the PDB code '7wwt', which already integrates the metal ions in the correct positions derived from the crystallographic structure, was downloaded.
- The PDB files of both proteins (3ecu and 7wwt) were opened, and the monomers other than the first one (marked as sequence A) were removed to obtain only one monomer for both of them.
- The residues of the two monomer sequences were aligned, calculating their similarity and identity.
- Consequently, the two monomeric structures were superimposed, and the root mean square deviation (RMSD) between the two structures was calculated.
- The monomer in 7wwt was removed, and the monomer in 3ecu with the metals derived from 7wwt was saved as wt.pdb.

The RMSD calculated according to MOE can be expressed as the RMSD of an array containing the RMSDs between each corresponding amino acid, using the following formulation:

$$RMSD_{MOE} = \sqrt{\frac{1}{M} \sum_{j=1}^M RMSD_j^2} \quad (1)$$

Where j represents the current residue and M represents the total number of residues of the protein.

The RMSD of a specific residue j can be written as:

$$RMSD_j = \sqrt{\frac{1}{N} \sum_{i=1}^N \delta_i^2} = \sqrt{\frac{1}{N} \sum_{i=1}^N |\vec{r}_i^{(1)} - \vec{r}_i^{(2)}|^2} = \sqrt{\frac{1}{N} \sum_{i=1}^N [(x_i^{(1)} - x_i^{(2)})^2 + (y_i^{(1)} - y_i^{(2)})^2 + (z_i^{(1)} - z_i^{(2)})^2]} \quad (2)$$

Where k represents the k -th atom in the residue j , x, y, z represent the coordinates, r represents the distances.

Generation of the mutant monomers

In order to obtain the mutated structures for comparison with the wild type, since they do not exist in the RCSB PDB database or at least not all of those of interest are present, it is necessary first to derive the amino acid sequence of the mutants and then obtain their structure in pdb format.

Generation of the mutated sequences

A MATLAB script (see [MATLAB script 1](#) in the scripts appendix) has been developed to modify an amino acid sequence within a fasta file by substituting a specific amino acid according to the specifications provided in a txt file. This script facilitates the generation of amino acid sequences for the selected mutants.

The input fasta sequence corresponds to the wild-type human SOD1 monomer, as extracted from the pdb file. This sequence is also available in the UNIPROT database under accession code 'P00441':

```
ATKAVCVLKGDPVQGIINFEQKESNGPVKVVWGSIKGLTEGLHGFHVHEFGDNTAGCTSA  
GPHFNPLSRKHGGPKDEERHVGDLGNVTADKDGVADVSIEDSVISLSGDHCCIIGRTLTVVHE  
KADDLGKGGNEESTKTGNAGSRLACGVIGIAQ
```

The text file, with its content outlined in table 5, contains line-by-line information necessary for the amino acid substitutions. This file is formatted as follows:

$$AA_1 \ P \ AA_2$$

where AA_1 represents the original amino acid, P denotes the position in the sequence, and AA_2 is the new amino acid that replaces AA_1 .

Table 5: Mutation codes

| | | |
|---|-----|---|
| G | 85 | R |
| A | 4 | V |
| I | 104 | F |
| G | 41 | D |
| G | 41 | S |
| D | 76 | V |
| G | 93 | A |

The output file consists of a number of fasta files equal to the number of mutants to be generated.

Obtaining the mutated monomeric structures in pdb format

Once the mutant sequences have been obtained, it is necessary to derive the protein structure in pdb format. For this purpose, the AI-based prediction tool AlphaFold2 was employed, which can be easily executed through a Python script on Google Colab [65]. Specifically, the parameters required included the input of a query sequence, i.e., the fasta sequence of the mutant whose protein structure is to be predicted (with any spaces removed), and setting the template mode to "custom," selecting the first monomer (chain A) from "3ecu" during the code execution. This procedure was repeated for each mutant, resulting in the acquisition of the structure for each one.

Obtaining the 'holo' variants of the mutated monomeric structures

The structures obtained previously are of the "apo" type, as they do not contain any metal complexes. Following the same protocol used earlier in the section "Obtaining the structure of the human wild-type holo protein," zinc and copper ions were added to each mutant structure to ensure protein stabilization.

Generation of the dimers

So far, the focus has been on generating the monomeric structures of the mutants of interest. However, to understand how prion-like conversion occurs and how a mutated SOD1 monomer may influence the stability of the wild-type SOD1 monomer when forming the dimeric structure (which is essential for the protein to fully execute its catalytic function), it is necessary to obtain the dimers. For this purpose, the Haddock 2.4 tool was utilized, which is capable of performing docking between large structures such as two proteins [66] [67] [68]. To ensure the correct upload of the monomers, the following modifications were made to the corresponding pdb files:

- The zinc and copper atoms were manually renamed from Zn and Cu to Zn²⁺ and Cu²⁺, respectively, while the associated residues were renamed from Zn and Cu to Zn2 and Cu2. The pdb file was then saved with these modifications.
- Using a tool from "pdbtools," accessible via the following link [69], the modified pdb file was uploaded as input. The pdb_selaltloc tool was added to the pipeline to clean the file of any alternate occupancy, and the process was run to produce a new pdb file.
- In the pdb file from the previous step, "ZN+" and "CU+" were appended with "2" to become "ZN2+" and "CU2+" and any following spaces were removed.

At this point, the pdb file of the monomer is ready to be correctly uploaded to Haddock for docking with other monomers.

This cleaning and formatting process of the pdb files to make them compatible with Haddock was performed for all previously saved mutated monomeric structures with integrated metal complexes using the manual method.

Specifically, docking was performed to generate the following dimers:

- wt+wt;
- wt+A4V;
- wt+G41D;
- wt+G41S;
- wt+D76V;
- wt+G85R;
- wt+G93A;
- wt+I104F;

where each "wt" refers to the wild-type monomeric structure, and A4V, G41D, G41S, D76V, G85R, G93A, and I104F refer to the mutated monomeric structures.

MD Simulations

To study the differences in conformation and properties between the proteins, once the dimers have been obtained using the Haddock software, molecular dynamics (MD) simulations are necessary. The software package used for this purpose is Amber.

Dimers preparation and simulation run

Simulations were conducted on the generated dimers. First, the structures were initially refined using the QuickPrep tool in MOE. This step was necessary to correct errors in the metal atom charges that

arose during monomer preparation for docking with Haddock and persisted in the resulting dimers. Additionally, the PDB files of the dimers were modified by changing their type from ME+2 to ME and the corresponding residue from ME2 to ME, where ME represents Zn or Cu ([see python script 1](#)). This modification was essential to ensure that zinc and copper atoms were recognized by tleap.

To ensure compatibility of the PDB files before processing them in tleap, the files were prepared using the following instructions. Those involve loading of course the necessary modules for using AMBER, cleaning the PDB file, and removing problematic hydrogen atoms not recognized by Amber (these hydrogens will be automatically re-added in tleap).

```
module load StdEnv/2023 gcc/12.3 openmpi/4.1.5 cuda/12.2 amber/22.5-23.5 # 1
Load the necessary modules for using Amber
```

```
pdb4amber -i dimer_c.pdb -o dimer_c_amber.pdb # 2 Converts the PDB file to be
readable by Amber
```

```
reduce -Trim dimer_c_amber.pdb > dimer_c_amber_noH.pdb # 3 Removes hydrogens
that Amber cannot handle (tleap will automatically add them)
```

After this initial preparation, tleap was used to make the structure ready for MD simulation by following the step-by-step protocol below:

- 1 Loading the ff19SB force field for the protein;
- 2 Loading the water model (TIP3P), it contains already the library file for bivalent ions (atomic_ions.lib);
- 3 Loading the parameterization files for bivalent ions (frcmod.ions1m_1264_tip3p);
- 4 Loading the previously prepared compatible PDB file;
- 5 Calculating the charge;
- 6 Solvating the system in a truncated octahedron box;
- 7 Adding Na^+ Cl^- ions to neutralize the structure, ensuring the total charge is zero;
- 8 Saving the topological parameters and coordinates of the new structure;
- 9 Saving the new structure.

The individual commands are provided below:

```
tleap # Launches the tleap tool for preparing molecular systems
source leaprc.protein.ff19SB # Loads the force field for proteins
source leaprc.water.tip3p # Loads the TIP3P water model
loadamberparams frcmod.ions2341m_1264_tip3p # Loads the parameter file for diva-
lent ions
dimer = loadpdb dimer_c_amber_noH.pdb # Loads the PDB file into the dimer varia-
ble
charge dimer # Calculates the net charge of the dimer
solvateOct dimer TIP3PBOX 10.0 # Sets a truncated octahedron with 10 angstrom
thickness as the solvent box
addIonsrand dimer Na+ #Na Cl- #Cl # Sets the number of ions to add to neutralize
the system's charge (see Matlab script to compute the number of ions to add)
saveAmberParm dimer dimer_c_amber_solv.prmtop dimer_c_amber_solv.inpcrd # Saves
the topology parameters and coordinates of the new structure
savepdb dimer dimer_c_amber_solv.pdb # Saves the new structure
quit # Exits the tleap program
```

To determine the exact amount of Na^+ and Cl^- ions needed to neutralize the charge as mentioned in step 7 of the previously outlined instructions, the following quick method introduced by Machado and his group [70] was used.

This method involves at first the calculation of the number of ions No for a concentration Co of 0.15M in a simulation box containing Nw water molecules, performed as

$$No = \frac{Nw \cdot Co}{56} \quad (3)$$

Then the number of positive and negative charges to add (and thus indirectly the number of Na^+ and Cl^- ions) is calculated as:

$$N^+ = Na^+ = No - \frac{Q}{2} \quad (4)$$

$$N^- = Cl^- = No + \frac{Q}{2} \quad (5)$$

and rounded up in case of odd Q .

Those operations were done using [MATLAB script 2](#).

In step 3, the parameterization files for bivalent ions were loaded to construct the 12-6-4 LJ-type nonbonded model within the AMBER force field for metal ion modeling. However, it is crucial to add the C4 term to the topology file. This can be achieved using *parmed*, a tool integrated into the *Amber* suite, by executing the following command:

```
parmed -i 1264_parmed.in -p dimer_c_amber_solv.prmtop
```

With the 1264_parmed.in file being:

```
loadRestrt dimer_c_amber_solv.inpcrd # Load the solvated coordinate file
setOverwrite True # Allow overwriting of existing files
add12_6_4 :CU watermodel TIP3P # Apply the 12-6-4 correction to Copper ions
using the TIP3P water model
add12_6_4 :ZN watermodel TIP3P # Apply the 12-6-4 correction to Zinc ions using
the TIP3P water model
outparm dimer_c_amber_solv_1264.prmtop dimer_c_amber_solv_1264.inpcrd # Save
the updated parameter and coordinate files
```

In this way, the topology and coordinate files required to perform the simulations for each dimer were obtained, namely:

```
dimer_c_amber_solv_1264.prmtop
dimer_c_amber_solv_1264.inpcrd
```

These simulations include a first step consisting of two minimization phases, with and without restraints respectively, in order to avoid steric clash errors and atomic overlap within the protein in the starting structure. This step was performed without the use of graphics cards, using the following instructions, where min1.in and min2.in are the input files for the minimization.

```

# minimization with restraints
pmemd -O -i min1.in -p dimer_c_amber_solv_1264.prmtop -c
dimer_c_amber_solv_1264.inpcrd -o min1.out -r min1.rst -ref
dimer_c_amber_solv_1264.inpcrd -inf min1.mdinfo

# minimization (full structure)
pmemd -O -i min2.in -p dimer_c_amber_solv_1264.prmtop -c min1.rst -o min2.out -r
min2.rst -inf min2.mdinfo

```

Once the minimized structure was obtained, the second step was performed, consisting of the heating, equilibration, and production simulation phases. The instructions for these steps are provided below, and they required the use of the `cuda` command for GPU utilization.

```

# run heating
pmemd.cuda -O -i nvt.in -o nvt.mdout -c min2.rst -r nvt.rst -x nvt_trajectories.nc
-inf nvt.mdinfo -p dimer_c_amber_solv_1264.prmtop -ref min2.rst

# run equilibration
pmemd.cuda -O -i npt.in -o npt.mdout -c nvt.rst -r npt.rst -x npt_trajectories.nc
-inf npt.mdinfo -p dimer_c_amber_solv_1264.prmtop -ref nvt.rst

# production run
pmemd.cuda -O -i md.in -o md.mdout -c npt.rst -r md.rst -x md_trajectories.nc -
inf md.mdinfo -p dimer_c_amber_solv_1264.prmtop -ref npt.rst

```

Analysis

At the end of the simulations, various types of analyses were performed using the `cpptraj` analysis tool included in the Amber software package. First, the time evolution of RMSD and the radius of gyration over time, as well as the RMSF for each protein residue, were calculated using the [CPPTRAJ script 1](#).

Subsequently, both the solvent accessible surface area (SASA) for each residue and for each frame of the protein of interest, as well as the total SASA of the protein per frame (calculated by summing the contributions of the various residues for each frame), were calculated using the [CPPTRAJ script 2](#).

Finally, with the same application, a conformational clustering was performed on the trajectory of each dimer, using the k-means algorithm to obtain 10 clusters ([see CPPTRAJ scripts 3](#)).

Candidate drugs: preparation for docking

Drugs were sought that could promote stabilization at the interface between the monomers constituting each dimer, thereby addressing the issue of misfolding and aggregation caused by prion-like behavior. One of these is a new inhibitor, a small molecule called PRG-A01 or Amisodin. In transgenic murine models with ALS, it can inhibit neuronal cell death caused by the overexpression of SOD1 mutants, inhibit aggregation between SOD1 and TDP-43, and has demonstrated therapeutic effects [71]. Since no structure of PRG-A01 in SDF or PDB format is available online, it was reconstructed from an image depicted in figure 38 [72] using the Chemical Sketch Tool from RCSB PDB, Marvin JS Editor [73].

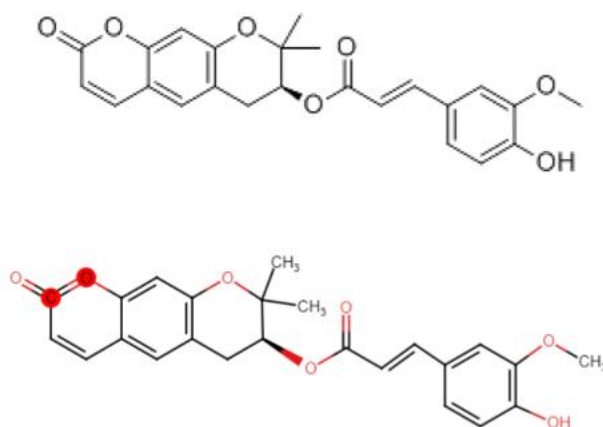


Figure 38. Comparison between the PRG-A01 structure online found and PRG-A01 structure draw with Marvin JS Editor.

Another candidate is Telbivudine, a synthetic nucleoside similar to thymidine that inhibits the replication of a specific virus associated with chronic hepatitis B infection. It is administered orally, has good tolerance, lacks toxicity, and poses no issues related to overdose. It has been shown to have a strong effect in preventing the misfolding and aggregation of SOD1 [74].

In addition to these, the drug S-XL6 was also considered. It was selected because co-crystallographic structures show the binding of S-XL6 to the cysteine residue 111, bridging between the monomers and thereby stabilizing the SOD1 dimer. Biophysical studies reveal that the degree of stabilization provided by S-XL6 (up to 24°C) is unprecedented for familial ALS and for any target protein of any kinetic stabilizer [75].

To expand the pool of ligands to be tested using the following docking procedures, the Chemical Structure Search tool from DrugBank was utilized, with a similarity threshold set to 0.6.

In this way, starting from PRG-A01, two additional ligands were identified: Rotenone and Epigallocatechin gallate.

Rotenone is an alkaloid, a naturally occurring insecticide and pesticide produced by several tropical plants, and it is also highly toxic to humans as it inhibits electron transport from Complex I to ubiquinone during cellular respiration [76], [77].

Epigallocatechin gallate (EGCG) is a polyphenol, specifically a type of catechin found in high concentrations in green tea and other plant-based beverages or foods [78]. It is investigated for the

treatment of hypertension and diabetic nephropathy and serves as a neuroprotective and antioxidant agent [79].

From Telbivudine, an additional 30 ligands were identified using the same tool (with the same similarity threshold), listed alphabetically below: TXS, 2",3"-Dideoxythymidine-5"-Monophosphate, 2",5"-Dideoxyuridine, 2"-Deoxyuridine, 2"-deoxy-5-(hydroxymethyl)uridine 5"-(dihydrogen phosphate), 2"-fluoro-5-ethylarabinosyluracil, 5"-deoxy-5"-piperidin-1-ylthymidine, 5-methyl-2"-fluoroarauracil F-18, Alovudine, Alovudine F-18, Brivudine, Brivudine monophosphate, Broxuridine, Clevudine, Ebselen, Edoxudine, Ethaselen, Floxuridine, Idoxuridine, Netivudine, Sorivudine, Thymidine-5"-Thiophosphate, TMP, Thymidine monophosphate, Thymidine-3",5"-Diphosphate, Thymidine-5"-Diphosphate, Thymidine-5"-(dithio)phosphate, Trifluridine, Uridine, Zidovudine.

A total of 35 ligands were selected, according to the previously specified criteria, and numbered in table 6. For each ligand, it was verified whether or not it complied with Lipinski's Rule of Five, information that is shown in the table.

Table 6: compounds library

| Ligand number | Ligand name | Molecular mass (g/mol) | LogP | H-bond donators | H-bond acceptors | Lipinski rule of 5 criteria met |
|---------------|---|------------------------|------|-----------------|------------------|---------------------------------|
| 1 | 1-(2,5-dideoxy-5-pyrrolidin-1-yl-beta-L-erythro-pentofuranosyl)-5-methylpyrimidine-2,4(1H,3H)-dione (TXS) | 291.33 | < 5 | 2 | 3 | Yes |
| 2 | 2",3"-Dideoxythymidine-5"-Monophosphate | 306.27 | -0.7 | 2 | 7 | Yes |
| 3 | 2",5"-Dideoxyuridine | 214.21 | -0.9 | 2 | 4 | Yes |
| 4 | 2"-Deoxyuridine | 228.23 | -1.0 | 2 | 4 | Yes |
| 5 | 2"-deoxy-5-(hydroxymethyl) uridine 5"-(dihydrogen phosphate) | 322.21 | -1.7 | 3 | 8 | Yes |
| 6 | 2"-fluoro-5-ethylarabinosyluracil | 262.2 | 1.4 | 2 | 5 | Yes |
| 7 | 5"-deoxy-5"-piperidin-1-ylthymidine | 281.3 | 1.5 | 2 | 5 | Yes |
| 8 | 5-methyl-2"-fluoroarauracil F-18 | 158.1 | -0.5 | 1 | 4 | Yes |
| 9 | Alovudine | 267.2 | 0.5 | 2 | 5 | Yes |
| 10 | Alovudine F-18 | 286.2 | 0.6 | 2 | 5 | Yes |
| 11 | Brivudine | 273.1 | 1.7 | 1 | 4 | Yes |
| 12 | Brivudine monophosphate | 353.2 | -1.1 | 3 | 8 | Yes |
| 13 | Broxuridine | 259.1 | 0.2 | 2 | 5 | Yes |
| 14 | Clevudine | 260.2 | 1.2 | 2 | 5 | Yes |
| 15 | Ebselen | 273.1 | 3.8 | 0 | 3 | Yes |
| 16 | Edoxudine | 243.2 | 1.0 | 2 | 5 | Yes |
| 17 | Epigallocatechin gallate | 458.4 | 1.0 | 8 | 11 | No |
| 18 | Ethaselen | 258.2 | 2.3 | 1 | 3 | Yes |
| 19 | Floxuridine | 246.2 | -0.8 | 2 | 6 | Yes |
| 20 | Idoxuridine | 354.1 | 1.1 | 2 | 6 | Yes |
| 21 | Netivudine | 271.2 | 1.4 | 1 | 5 | Yes |
| 22 | PRG-A01 | 420.0 | 4.1 | 1 | 7 | Yes |
| 23 | Rotenone | 394.4 | 4.2 | 0 | 7 | Yes |
| 24 | S-XL6 | 122.0 | 1.6 | 0 | 0 | Yes |
| 25 | Sorivudine | 244.1 | 1.5 | 2 | 5 | Yes |
| 26 | THYMIDINE-5"-THIOPHOSPHATE | 366.2 | -1.5 | 3 | 9 | Yes |
| 27 | TMP (Thymidine Monophosphate) | 322.2 | -1.3 | 3 | 7 | Yes |
| 28 | Telbivudine | 242.1 | 1.5 | 2 | 5 | Yes |
| 29 | Thymidine monophosphate | 322 | <<5 | 3 | 9 | Yes |
| 30 | Thymidine-3",5"-Diphosphate | 402.2 | -2.0 | 3 | 10 | No |
| 31 | Thymidine-5"- Diphosphate | 402.2 | -2.0 | 3 | 10 | No |
| 32 | Thymidine-5"-(dithio)phosphate | 354.2 | -1.5 | 3 | 7 | Yes |
| 33 | Trifluridine | 296.2 | 0.6 | 2 | 6 | Yes |
| 34 | Uridine | 244.2 | -2.1 | 4 | 6 | Yes |
| 35 | Zidovudine | 267.2 | 0.5 | 2 | 6 | Yes |

Virtual screening with docking procedure

To evaluate which of the candidate ligands is the most effective at stabilizing the interface between the two monomers constituting the dimer and to conduct virtual screening, docking procedures were performed using AUTODOCK VINA. This software reports a value for each ligand-protein pair that is proportional to their binding affinity or free energy of binding (ΔG), calculated using an appropriate scoring function and measured in kcal/mol. A more negative value indicates a higher free energy of binding and thus a greater likelihood that the ligand will interact strongly with the protein.

Since no information is available about the binding site of the ligands from co-crystallographic structures (except for S-XL6 in the structure 8CCX found in RCSB), AUTODOCK VINA was used to perform a purely blinded docking, where the whole protein was selected as a potential binding site, a procedure that requires AUTODOCK TOOLS software.

AUTODOCK VINA blind docking

The protocol is as follows:

- Convert each PDB file with the dimer into a PDBQT file:
 1. Remove water if necessary;
 2. Add hydrogens if necessary;
 3. Add charges;
 4. Save the file as PDBQT and set manually with a text editor the correct charges for zinc and copper;
- Choose a gridbox to encompass the entire protein as a potential binding site and create an config.txt file as shown in the [AUTODOCK VINA configuration file](#).

The parameters `size_x`, `size_y`, and `size_z` are used to set the dimensions of the grid for calculating the possible spatial conformations of the ligand during the docking procedure. The grid spacing parameter also significantly affects the grid size but is not included in the configuration file because, in AUTODOCK VINA, it is automatically set to 1 Å. In AUTODOCK TOOLS, however, it is set by default to 0.375 Å but is adjustable by the user. Therefore, in AUTODOCK TOOLS, it is necessary to set the grid spacing to 1 Å to ensure consistency with the default setting in VINA, and then adjust the `size_x`, `size_y`, and `size_z` parameters accordingly to encompass a volume containing the entire protein. The `num_modes` parameter specifies the number of poses to be retrieved, which, after docking, will be ranked from best to worst. The `energy_range` is the maximum energy difference between the best and the worst binding modes, expressed in kcal/mol. The parameter `num_mode` indicates how many modes or poses of the ligand of interest to generate, in this case 10, which are ranked by VINA based on the energy affinity value.

- Convert each ligand pdb file (obtained by saving sdf files as pdb through MOE after performing a QuickPrep) into a pdbqt file.
- Use a [.bat script, available in the script appendix](#), to perform multi-ligand docking with AUTODOCK VINA for each dimer, iterated a desired number of times, in this case, 10, to ensure greater consistency and to verify whether the affinity value remains relatively constant across different iterations and if the binding site location varies a lot across the iterations.

Docking was performed for all ligands and dimers, generating 100 poses for each ligand, as 10 models were used per run (with `num_mode` = 10), and the process was repeated for 10 runs. A significant variability was observed in the poses of a given ligand, both in terms of energy affinity and the

localization of binding sites on the protein, influenced by the blind nature of the docking. Therefore, it was evaluated whether the ligand showed a preference for specific regions of the protein, identifying any preferential binding sites. To this end, a clustering of the poses based on RMSD was performed, with a threshold value of 4 Å to group similar poses, allowing for the identification of persistent binding regions and the assessment of ligand specificity. For the RMSD calculation, a Python script was used to analyze the PDBQT files of the poses and compute the RMSD between all 100 poses, producing a square $N \times N$ matrix, where N is the number of poses, and each element represents the RMSD between the pose in the i -th row and the pose in the j -th column. The matrix is saved in a text file. The script is available as [Python script 2](#) in the appendix.

Given a dimer, once the RMSD matrices for each ligand are obtained, they can be loaded as input files into the [MATLAB script 3](#), which applies pose clustering for each ligand based on the RMSD value and provides an output visualization of the results.

Results and discussion

Structure of the human wild-type holo protein

The monomeric human SOD1 structure derived from "3ecu" (chain A) and the animal one derived from "7wwt" (chain A) exhibit, after alignment of the amino acid sequence, an identity of 81.7% and a similarity of 88.8%. Additionally, the RMSD of the two structures after superposition, shown in figure 39, is 0.526 Å. The fact that the structures are so similar implies that the location of metal ions in the animal protein can be used, after the superposition, to locate the same ions in the human structure.

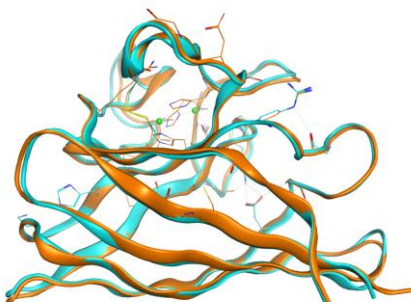


Figure 39. The monomer derived from "3ecu" is shown in light blue, superimposed on the monomer derived from "7wwt" in orange, with the metal ions depicted in green.

Mutant monomers generation

Mutated sequences

The output file from the MATLAB code, containing the monomer sequences in FASTA format, is available below. The point mutations are highlighted in yellow in each sequence, in the following table 7:

Table 7: mutated aminoacidic sequences derived from wild type sequence.

| | |
|-------|--|
| wt | ATKAVCVLKGDGPVQGIINFEQKESNGPVK VWGSIKGLTEGLHGFHVHEFGDNTAGCTSAGPHFNPLSRK HGGPKDEERHVGDLGNVTADKDG VADVSIEDSVISLSGDHCIIGRTL VVHEKADDLGKGGNEESTKTGNA GSRLACGVIGIAQ |
| A4V | ATKAVCVLKGDGPVQGIINFEQKESNGPVK VWGSIKGLTEGLHGFHVHEFGDNTAGCTSAGPHFNPLSRK HGGPKDEERHVGDLGNVTADKDG VADVSIEDSVISLSGDHCIIGRTL VVHEKADDLGKGGNEESTKTGNA GSRLACGVIGIAQ |
| G41D | ATKAVCVLKGDGPVQGIINFEQKESNGPVK VWGSIKGLTEGLHGFHVHEFGDNTAGCTSAGPHFNPLSRK HGGPKDEERHVGDLGNVTADKDG VADVSIEDSVISLSGDHCIIGRTL VVHEKADDLGKGGNEESTKTGNA GSRLACGVIGIAQ |
| G41S | ATKAVCVLKGDGPVQGIINFEQKESNGPVK VWGSIKGLTEGLHGFHVHEFGDNTAGCTSAGPHFNPLSRKH GGPKDEERHVGDLGNVTADKDG VADVSIEDSVISLSGDHCIIGRTL VVHEKADDLGKGGNEESTKTGNAG SRLACGVIGIAQ |
| D76V | ATKAVCVLKGDGPVQGIINFEQKESNGPVK VWGSIKGLTEGLHGFHVHEFGDNTAGCTSAGPHFNPLSRK HGGPKDEERHVGDLGNVTADKDG VADVSIEDSVISLSGDHCIIGRTL VVHEKADDLGKGGNEESTKTGNA GSRLACGVIGIAQ |
| G85R | ATKAVCVLKGDGPVQGIINFEQKESNGPVK VWGSIKGLTEGLHGFHVHEFGDNTAGCTSAGPHFNPLSRK HGGPKDEERHVGDLGNVTADKDG VADVSIEDSVISLSGDHCIIGRTL VVHEKADDLGKGGNEESTKTGNA GSRLACGVIGIAQ |
| G93A | ATKAVCVLKGDGPVQGIINFEQKESNGPVK VWGSIKGLTEGLHGFHVHEFGDNTAGCTSAGPHFNPLSRK HGGPKDEERHVGDLGNVTADKDG VADVSIEDSVISLSGDHCIIGRTL VVHEKADDLGKGGNEESTKTGNA GSRLACGVIGIAQ |
| I104F | ATKAVCVLKGDGPVQGIINFEQKESNGPVK VWGSIKGLTEGLHGFHVHEFGDNTAGCTSAGPHFNPLSRK HGGPKDEERHVGDLGNVTADKDG VADVSIEDSVISLSGDHCIIGRTL VVHEKADDLGKGGNEESTKTGNA GSRLACGVIGIAQ |

Mutated structures

The mutated structures predicted with AlphaFold2 were superimposed onto the wild-type structure (which was also used as a template), and the RMSD was calculated for each combination. This is visualized in figure 40 both for each residue and as a single value (according to the MOE RMSD definition, defined in the methods). The maximum RMSD calculated according to the MOE definition (related to the G85R mutant) is 0.4511 Å, indicating that the mutant PDB structures do not show significant differences compared to the wild-type structure, as the mutations considered are point mutations, affecting only a single amino acid. However, the subsequent MD simulations performed on dimers are intended to determine whether these mutations may lead to conformational and properties differences.

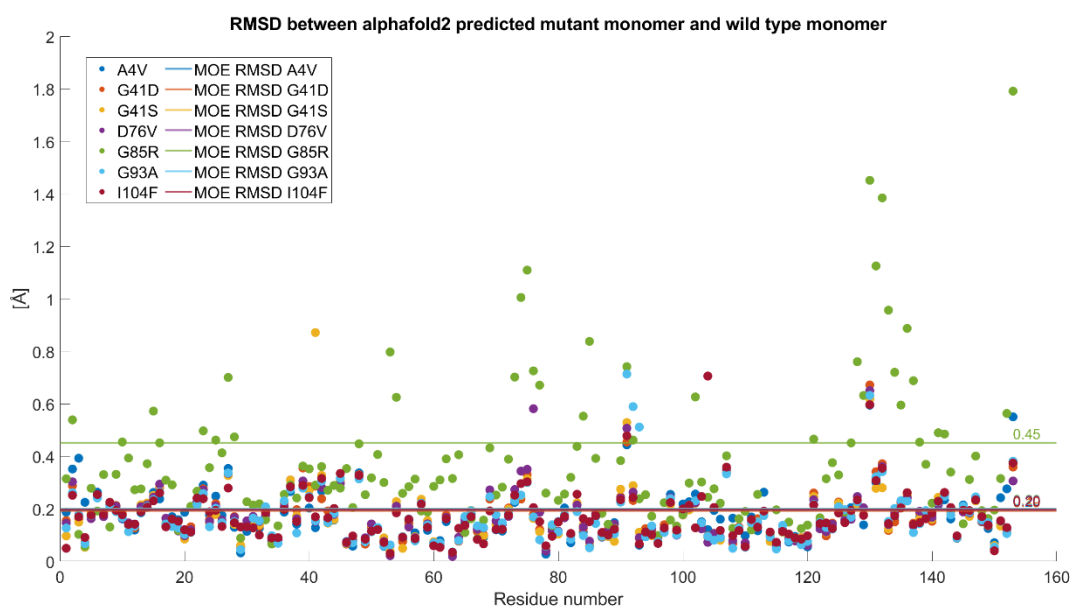


Figure 40. A comparison is shown between the RMSD of the various mutants (visible as dots in the legend) based on their position in the amino acid sequence. The horizontal lines, following the same color scheme, represent the RMSD values calculated according to MOE for each mutant.

Dimers generation

Dimer interface characteristics

The dimer generation was performed through the protein-protein docking software Haddock. Dimerization is a process where two monomeric structures bind together thanks primarily to hydrogen bonds in order to build a more complex and in case of SOD1 more functionally efficient and stable structure called dimer indeed (fig. 41).

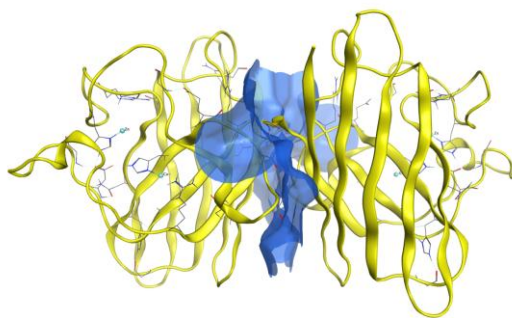


Figure 41. A generic SOD1 dimer of the builded ones is shown in yellow with the surface which include the contacts between the two monomers enlightened in blue.

The eight generated dimers shared quite the same behavior during the dimerization process, so that figure 41 can be representative of all the dimers. From a theoretical point of view the residues implicated in the dimerization process should be the following: 50-53, 114, 148, 150-153 [80]. Figure 42 let us visualize for each dimer, after the protein-protein docking process, which residues are implicated in the hydrogen bonds between chain A (first monomer) and chain B (second monomer). It's clear that most of the dimers share the same hydrogen bonds, even though dimers wt+G41D, wt+G41S and wt+D76V have another hydrogen bond, the latter, in a different position.

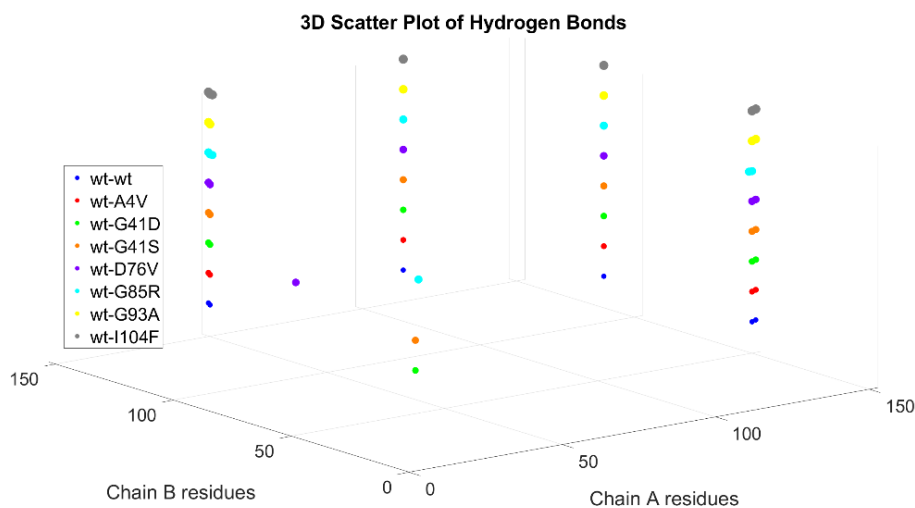


Figure 42: representation of which pairs of residues from chain A and chain B are involved in the hydrogen bonds at the interface of the various dimers.

Ramachandran plots

Figure 43 shows the Ramachandran plots comparing wild type (wt+wt) dimers with various mutants, highlighting the variations in the torsional angles phi (ϕ) and psi (ψ) of the mutated residues. The structures used are the ones generated via AlphaFold and Haddock. In the comparison with wt+G41D, the green dot of residue A4 shifts significantly from (131, -162) to (80, -166), indicating a major conformational change, especially in the phi angle. Similarly, in wt+G41S, the same residue A4 shifts even more drastically from (131, -162) to (175, -172), suggesting that both mutations G41D and G41S cause significant changes in the torsional angles, with important conformational effects but in different directions. In the case of wt+G85R, the red dot of residue G85 shows a marked shift from (121, -129) to (-174, -172), with a complete reversal of the phi angle and a large variation in the psi angle, indicating a notable perturbation in the local structure. These three mutations, G41D, G41S, and G85R, exhibit the most evident variations in the comparisons between wild type and mutated, with significant conformational changes that could profoundly affect the structure and functionality of the SOD1 dimer.

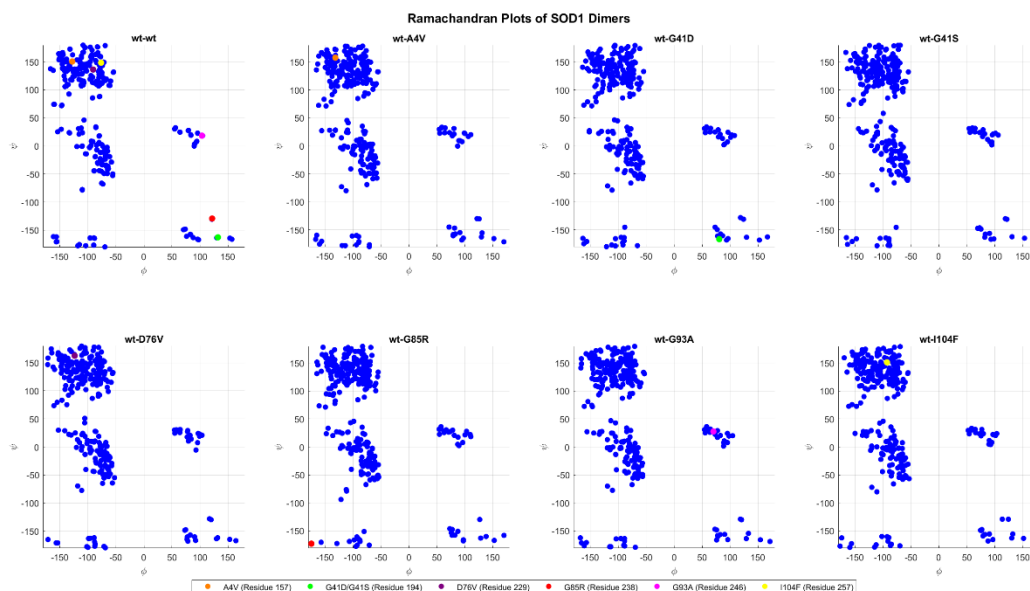


Figure 43: Ramachandran plot of protein residues for all dimers. Each blue point represents the pair of ϕ and ψ values for a specific generic residue, while colored points represent the same but for a mutated residue.

MD simulations: data analysis

RMSD

The graph in figure 44 shows that, excluding the first 100 ns, the wild-type dimer (wt+wt) maintains a relatively stable RMSD around 2 Å for about 500 ns, with a slight increase to around 3 Å in the second half of the simulation, suggesting overall stability with moderate structural fluctuations. The A4V mutation exhibits an RMSD that, after the first 100 ns, oscillates between 4 and 6 Å, with greater structural instability persisting throughout the remainder of the simulation, indicating a significant impact of the mutation. For the G41D mutation, the RMSD stabilizes around 4 Å after 100 ns and shows minimal fluctuations until the end of the simulation, indicating a relatively stable structure after an initial period of instability. The G41S mutation also stabilizes around 3 Å after the first 100 ns, showing moderate stability and contained fluctuations. The D76V mutation displays a similar trend, with the RMSD oscillating around 2.5 Å after the initial 100 ns, suggesting good structural stability without significant variations. In the case of the G85R mutation, the behavior is similar to that of D76V, with the RMSD remaining stable around 3 Å after 100 ns, showing moderate fluctuations but no significant structural changes. The G93A mutation shows very stable behavior, with a constant RMSD of about 3 Å throughout the simulation, without major fluctuations. Finally, the I104F mutation also presents a stable RMSD around 3 Å after the first 100 ns, suggesting a globally stable structure. Overall, the A4V mutation stands out for its greater instability and fluctuations over time, while the other mutations (G41D, G41S, D76V, G85R, G93A, I104F) display similar behaviors with RMSD values between 2.5 and 4 Å and structural stability that remains consistent beyond 100 ns.

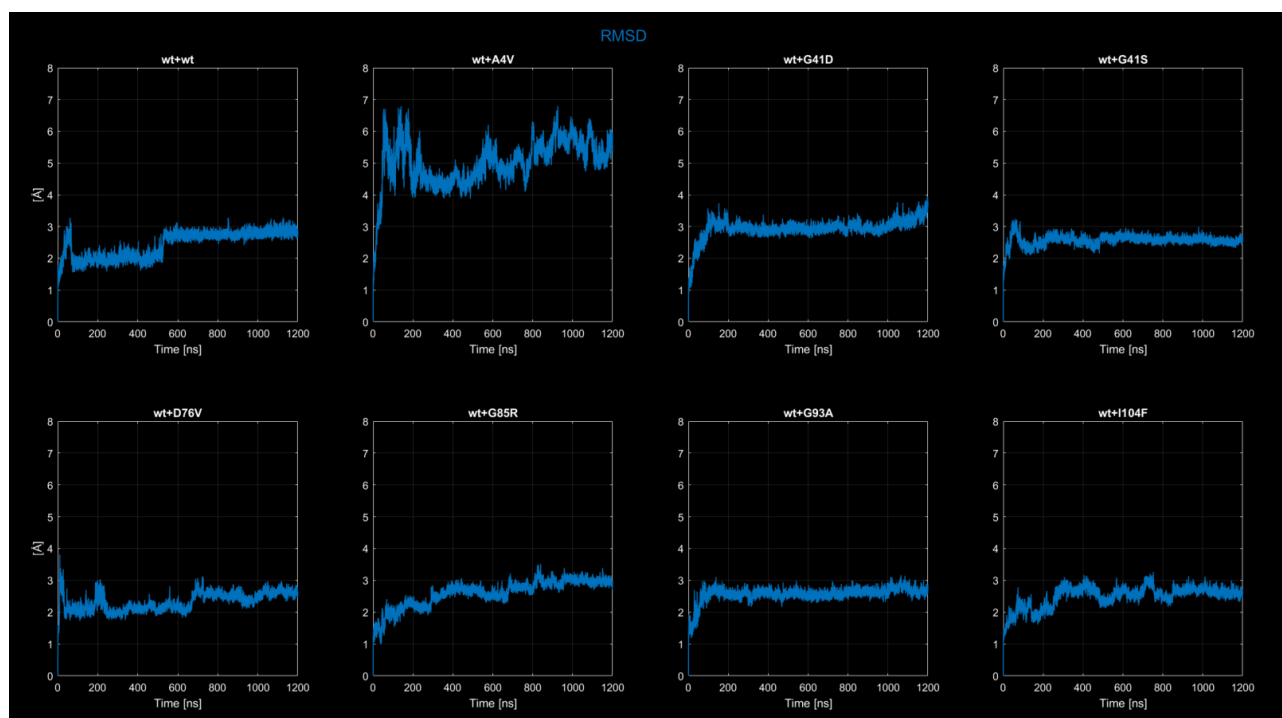


Figure 44: RMSD time evolution across all dimers.

Radius of gyration

Figure 45 gives a view of the time evolution of the radius of gyration of all dimers. After the first 100 ns, the wild-type dimer (wt+wt) shows a relatively stable Rg , fluctuating between 19.6 Å and 20.2 Å for most of the simulation, with a slight increase after 600 ns, suggesting a compact structure with a mild expansion over time. The A4V mutation exhibits similar fluctuations, ranging from 19.6 Å to 20.4 Å, with an initial decrease in compactness followed by slightly more pronounced fluctuations compared to the wild type, indicating a minimal impact on overall compactness. In the case of the G41D mutation, the Rg quickly decreases to about 19.3 Å after the initial 100 ns, remaining stable and indicating greater compactness compared to the wild type. The G41S mutation also shows a reduction in Rg to around 19.3 Å, maintaining stability, suggesting a more compact structure. The D76V mutation, similar to wt+wt, shows fluctuations between 19.6 Å and 20.2 Å, with a slight increase in Rg after 600 ns, maintaining a stable and compact structure. The G85R mutation follows a similar trend, with fluctuations between 19.6 Å and 20.2 Å, suggesting a stable and compact structure without significant variations in compactness. The G93A mutation shows a reduction in Rg to about 19.3 Å, followed by sustained stability, indicating greater structural compactness similar to G41D and G41S. Finally, the I104F mutation shows fluctuations between 19.6 Å and 20.2 Å, with a slight increase in Rg after 600 ns, maintaining an overall stable and compact structure. In summary, the mutations G41D, G41S, and G93A induce greater structural compactness, while the other mutations (A4V, D76V, G85R, I104F) and the wild type maintain a stable and compact structure with similar fluctuations.

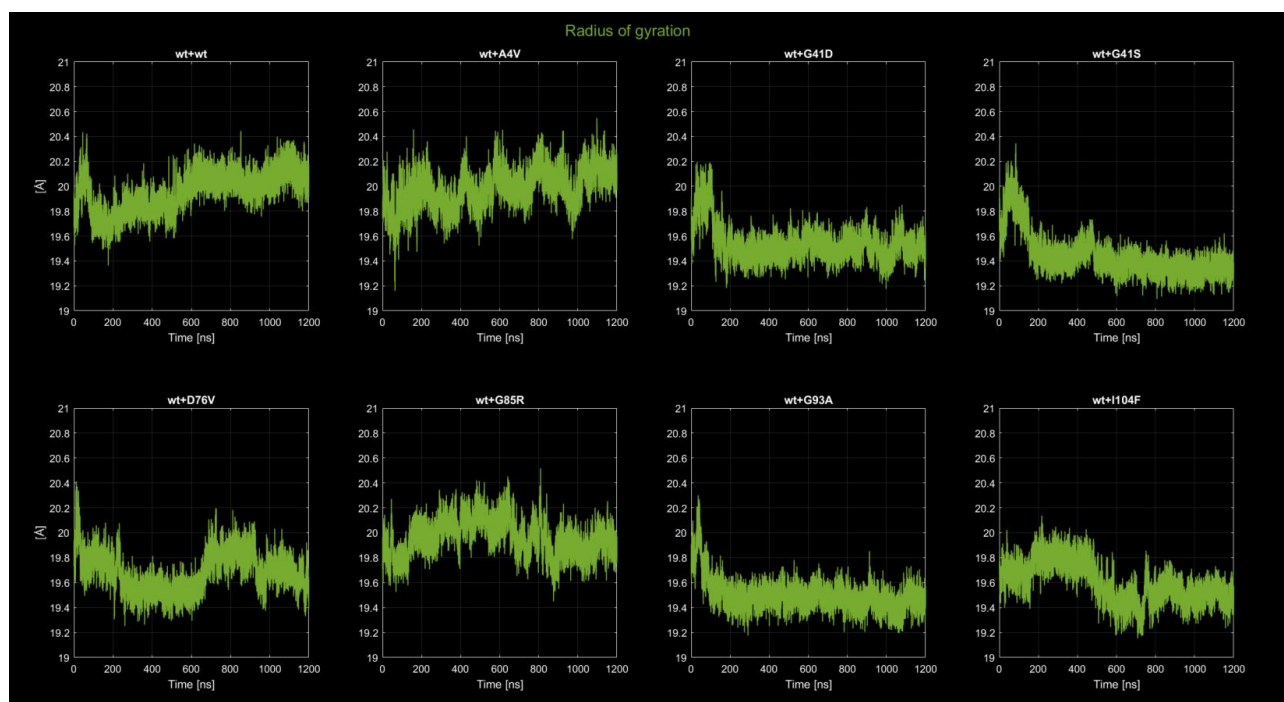


Figure 45: radius of gyration time evolution across all dimers.

SASA

Figure 46 shows the SASA as a function of time (ns) for the eight different dimers of the SOD1 protein. The vertical axis represents the SASA in \AA^2 , and the horizontal axis represents time in nanoseconds. In the wt+ wt graph, the SASA remains fairly constant over time, with fluctuations between 1.25 and $1.4 \times 10^4 \text{\AA}^2$. In the case of the wt+A4V dimer, the fluctuations are broader compared to wt + wt, with values oscillating between 1.25 and $1.45 \times 10^4 \text{\AA}^2$. The wt+G41D graph shows a steadily decreasing SASA, going from about 1.45 to $1.25 \times 10^4 \text{\AA}^2$ in 400 ns and maintaining the last value constant until the end of the simulation. Similar to G41D, the wt+G41S graph shows a decrease in SASA from about 1.35 to $1.2 \times 10^4 \text{\AA}^2$, indicating that the G41S mutation may also reduce solvent exposure over time. The wt+D76V graph shows a stable SASA, fluctuating between 1.25 and $1.4 \times 10^4 \text{\AA}^2$. The wt+G85R graph shows a slightly increase and irregular fluctuations but with a SASA mostly ranging between 1.25 and $1.4 \times 10^4 \text{\AA}^2$. The wt + G93A graph shows a decrease in SASA from about 1.35 to $1.2 \times 10^4 \text{\AA}^2$. The wt+I104F graph shows a fluctuation within the first 600 ns.

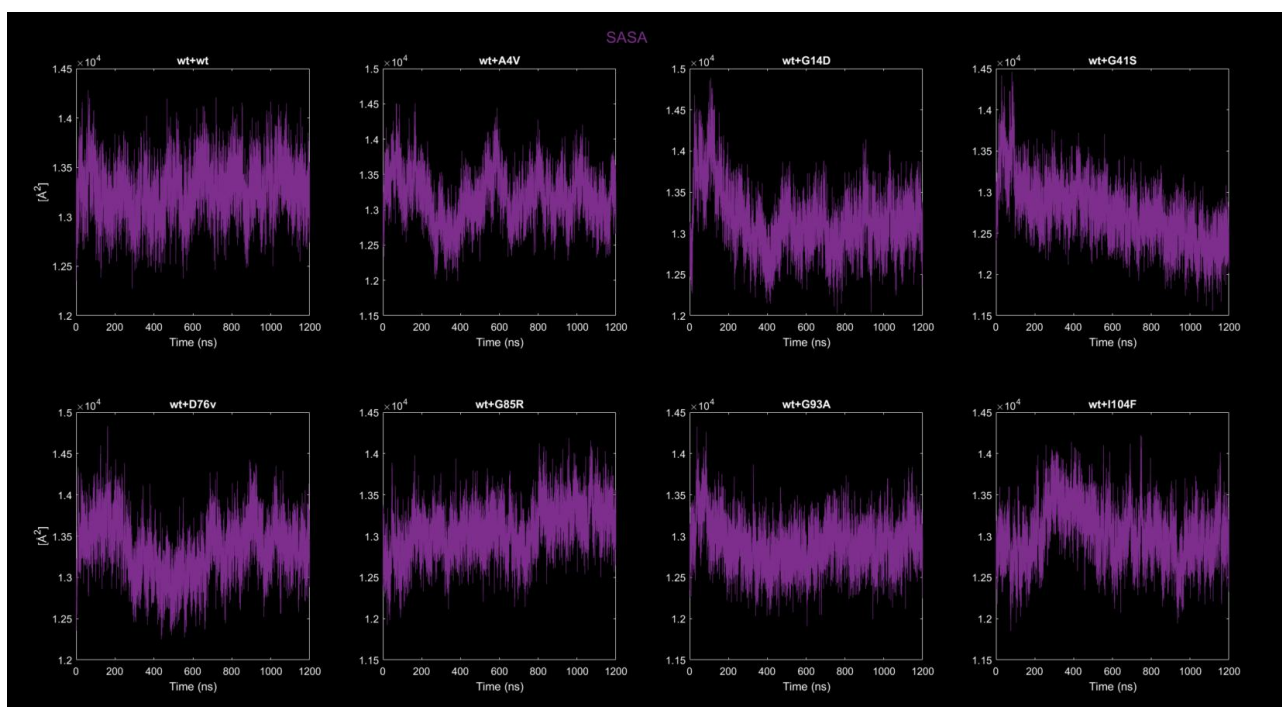


Figure 46: SASA time evolution across all dimers.

RMSF

The graphs displayed in figure 47 show the RMSF (Root Mean Square Fluctuations) relative to the number of residues for various protein dimers, with the wild type dimer (wt+wt) used as a control to compare the variations induced by mutations. Each dimer consists of two monomers, the first colored in red and the second in yellow. In each graph, the green dot represents the mutated residue, located in the second monomer. Regarding the wt+wt dimer, two major peaks are observed near the end of each monomer, specifically at positions 135 and 286, with values just below 6 Å and above 10 Å. In the wt+A4V dimer, the first peak identified in the wild-type protein at the end of the first monomer is absent. In the mutated monomer, residues 280 to 289 form a single jagged peak of more than 5 Å, and an additional peak formed by residues 225-227 can also be observed, corresponding to the second peak in wt+wt, with a lower maximum value of 9 Å. Regarding the wt+G41D dimer, the residues forming peaks are from 129 to 134 towards the end of the first monomer, from 221 to 227, and from 281 to 291. It is noted that the first and third peaks have lower values compared to the wild type, while the second is more intense, reaching almost 5 Å (in wt+wt, the peak in this position was not mentioned due to its low intensity). The wt+G41S dimer does not show significant peaks except for the one at the end of the second monomer, formed by residues 281 to 293, which also appears in wt+G41D, with similar values. In the wt+D76V dimer, the only significant peak, in a position similar to the first peak of wt+wt, is formed by residues 129 to 136, with a value exceeding 6 Å. The wt+G85R dimer shows three peaks in the same regions as wt+G41D. wt+G93A does not show any significant peaks, while wt+I104F presents a situation similar to wt+wt, with the second peak being less pronounced and consisting of more residues.

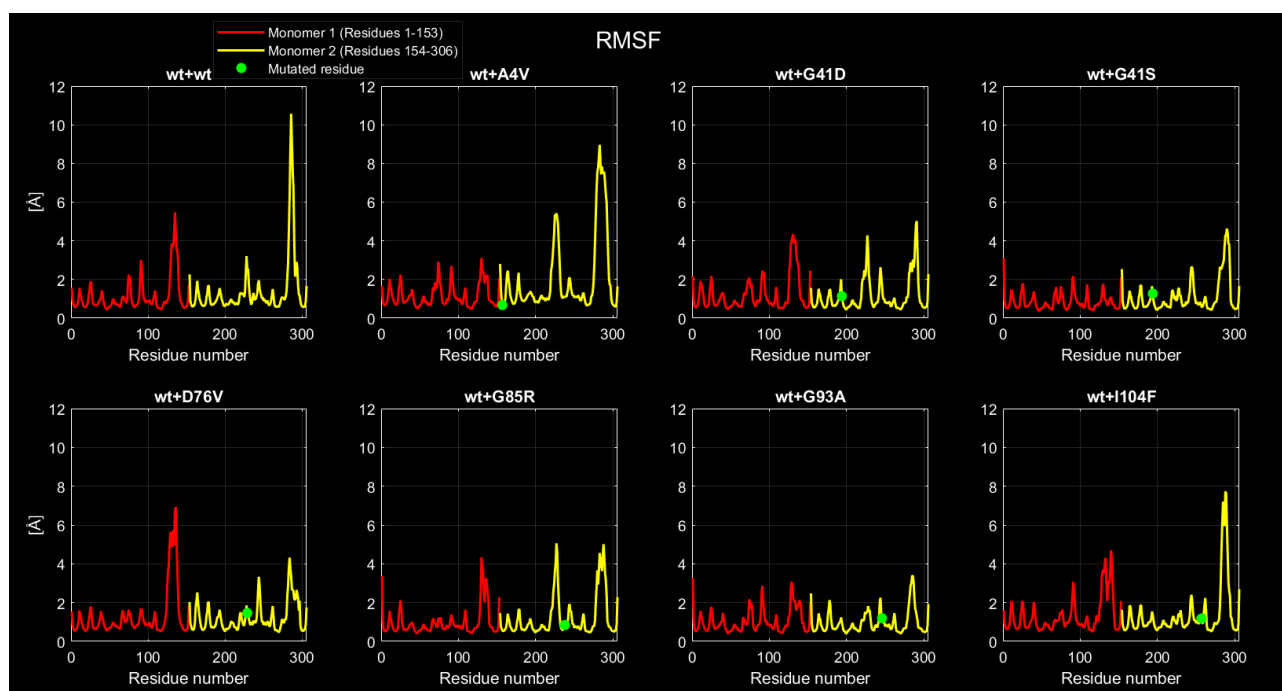


Figure 47: RMSF vs residues, for each dimer

The image visualizable in figure 48, showing the overlaid RMSF curves for the various dimers, including the mutated ones, provides a direct comparison of residue fluctuations among the different proteins. The colored curves represent the different dimers with specific mutations and allow observation of how each mutation influences the dynamics of the dimer compared to the wild type (wt+wt), which is shown in blue.

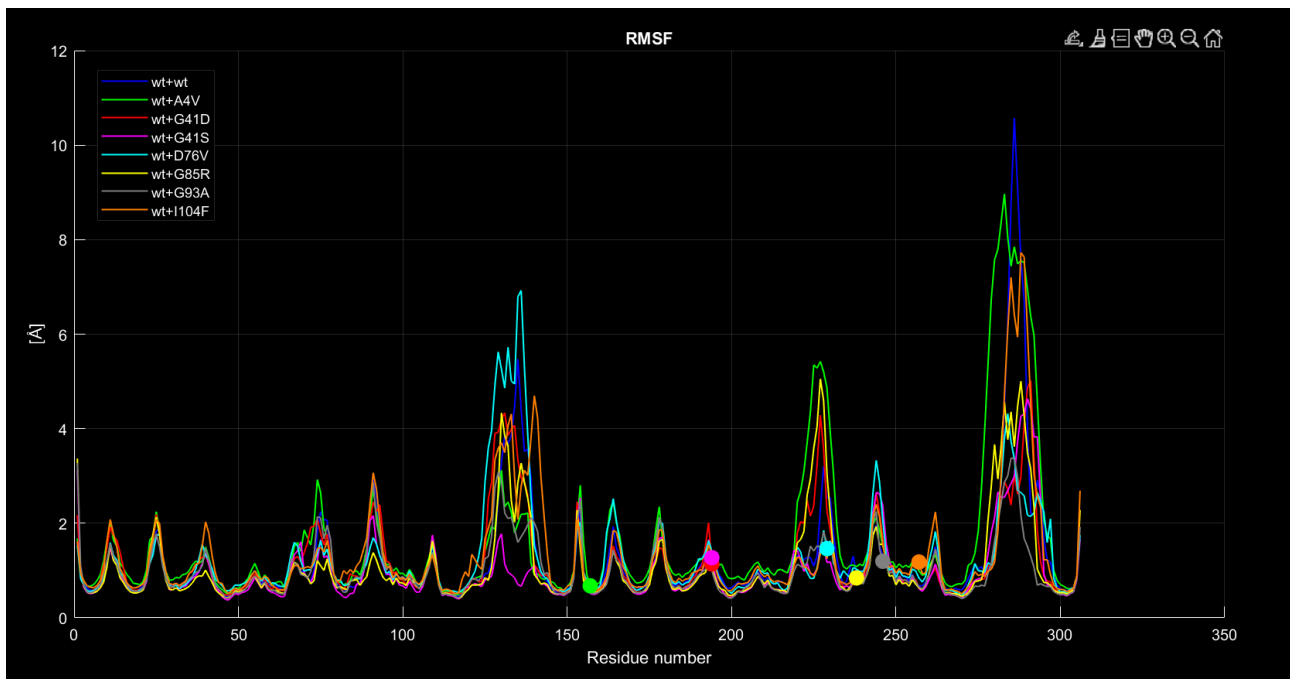


Figure 48: same as figure 47, but with a different visualization: each dimer RMSF is plotted in the same graph.

Conformational clustering

Figure 49 shows the clusters to which the frames of the trajectory from each MD simulation belong, for each dimer. It can be observed that after conformational clustering, 10 clusters were found for each protein. As the number of frames increases during the simulation, the frames transition into different conformations, and thus different clusters. The clusters are numbered from 0 to 9, ranked from the most to the least populous. Although the red cluster identified as 0 is the largest, possibly containing the most frames and therefore being the most recurring and representative of the protein structure, it is not the final cluster in all dimers -i.e., the one containing the last frames of the protein. In fact, there is some variability among the dimers regarding the distribution of frames across the different clusters. However, it is difficult to attribute this variability to the mutation. Additionally, the k-means algorithm was applied, which requires prior knowledge of the number of clusters, set to 10 in this case. A clustering algorithm that does not require prior knowledge of the number of clusters can be applied in future studies to verify whether the dimers adopt a larger number of conformations compared to the wild type.

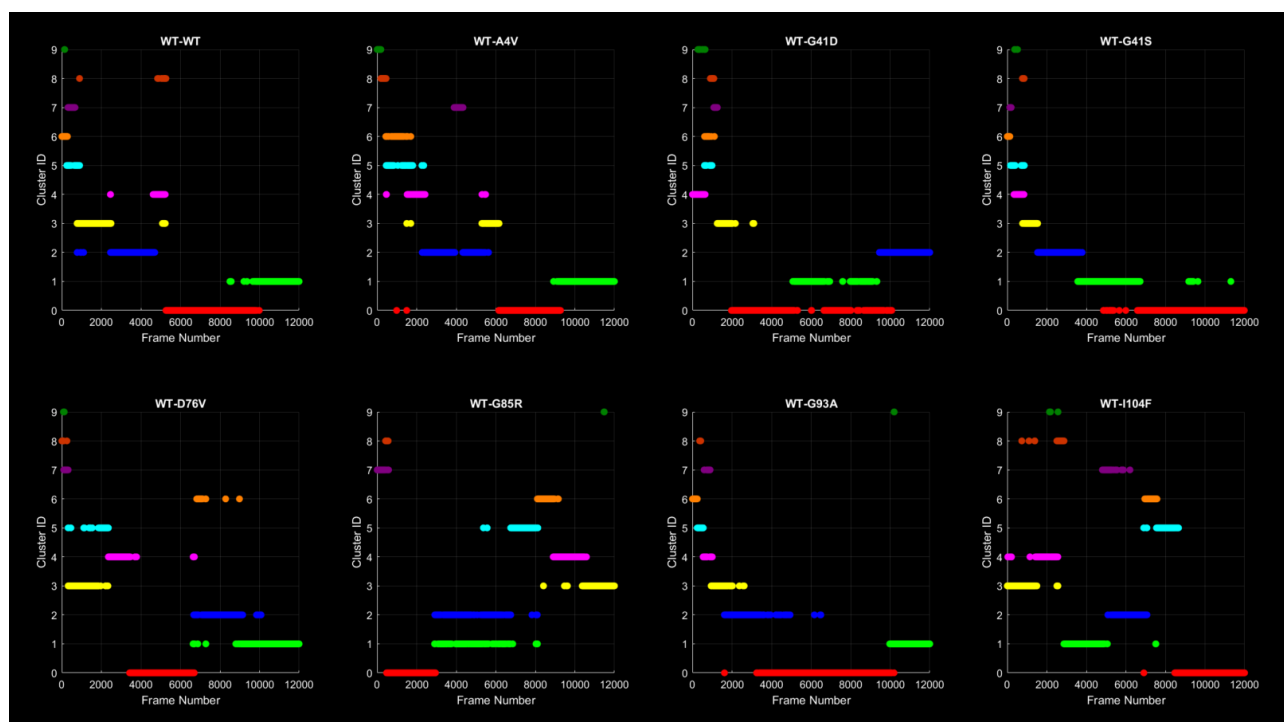


Figure 49: Cluster population over time. Each trajectory frame is assigned to a specific cluster, represented by a distinct color. It is important to note that, although the same color scheme is used, a cluster of a given color in one protein does not correspond to the cluster of the same color in another protein.

In Figure 50, the structures corresponding to the various clusters of the fully wild-type dimer (wt+wt) are shown, colored differently compared to the previously presented graph and overlaid. Visually, it can be observed that the orange structure representing cluster 5 and the pale pink structure corresponding to cluster 7 displays alpha helices that deviate from those in other structures, indicating the particular conformations this type of protein can adopt.

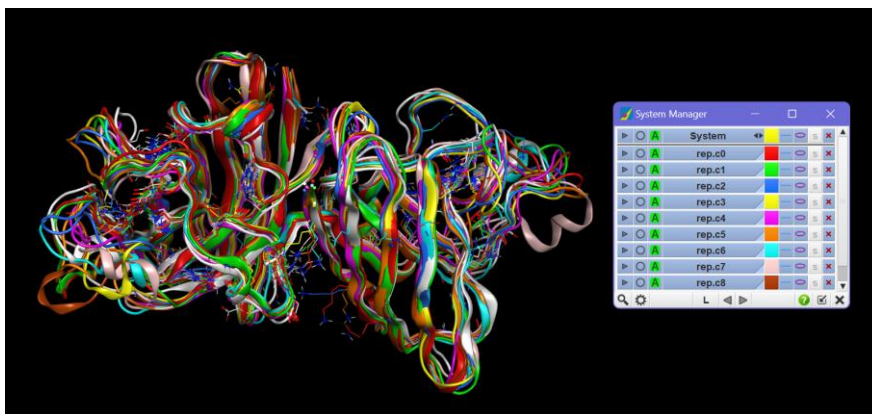


Figure 50: superposition of wt+wt dimer representative structures, each color corresponding to a specific cluster.

Ramachandran plots

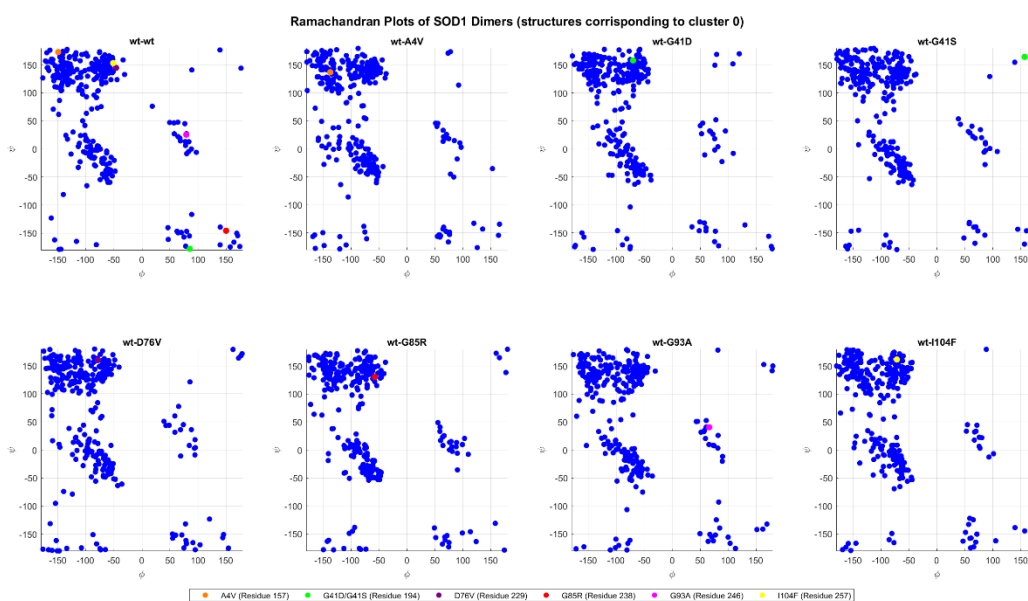


Figure 51. It shows the Ramachandran plots for the SOD1 protein dimers, both in its wild-type (wt+wt) form and in various mutated variants. The blue dots show the pairs of torsional angles ϕ and ψ for all residues in the structure, while the colored dots highlight the specific residue which is subjected to a mutation in each dimer. The structures used to generate these plots are derived from conformational clustering of molecular dynamics simulations, with the most representative structures chosen from the largest clusters (cluster 0 for all dimers).

From figure 51 it's clear to see that the mutations represented by the colored dots show significant differences in torsional angles compared to the wt+wt form. For example, in the wt+G41D mutation (green dot), the mutated residue has coordinates $(-70, 158)$ compared to $(85, -177)$ in the wt+wt, indicating a significant change that could destabilize the local structure. Similarly, in the wt+G41S mutation (another green dot), large differences are observed, with coordinates $(156, 164)$ compared to $(85, -177)$ in the wt+wt, highlighting a significant alteration in angular conformation. Another mutation with a strong impact is wt+G85R (red dot), with coordinates $(-57, 130)$ compared to $(150, -145)$ in the wt+wt, suggesting a relevant change in the local secondary structure. In the wt+A4V mutation (orange dot), the coordinates $(-136, 137)$ compared to $(-149, 172)$ in the wt+wt show a variation in ϕ with a slight difference in ψ , potentially affecting the orientation of the structure.

Virtual screening with docking procedure

AUTODOCK VINA blinded docking, affinity evaluation and pose clustering

The graphs in the following images are similar scatter plots with colored data points, visualizing molecular docking results. Each graph, one for each dimer, displays:

- X-axis: ligands, numbered sequentially.
- Y-axis: cluster sizes (number of poses) for each ligand.
- Color: the color of the points reflects the average Vina score for each cluster, following a color scale ranging from dark blue (more favorable scores, around -8 kcal/mol) to red (less favorable scores, around -3 kcal/mol).

These graphs provide a clear representation of the distribution of poses and their binding affinities for each ligand, with the colors allowing a quick evaluation of the average Vina scores for each cluster.

Regarding the docking of the wt+wt dimer (fig. 52) , it emerges that the ligands with the highest affinity, where affinity is understood as binding free energy, are ligands 5, 10, 22, 23, 28, and 29. Ligand 24, despite having low affinity, is one of those, along with ligands 8, 13, and 34, that presents a large cluster of poses, with more than 30 poses.

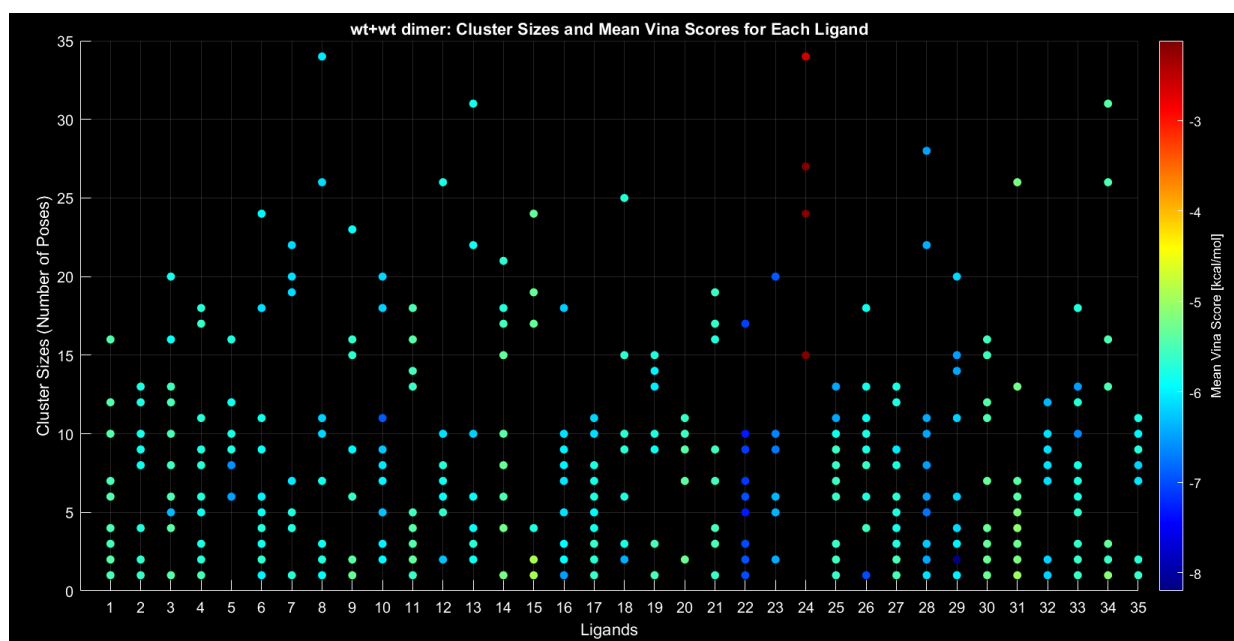


Figure 52: wt+wt dimer clustering results

As for the docking with the mutated wt+A4V dimer (fig. 53), it shows a similar behavior, with ligand 24 still performing poorly in terms of affinity values, but one of its clusters reaches a pose count of more than 60. The ligands with the highest affinity are 2, 16, 19, 22, 23, 28, 29, 32, and 35, among which ligands 22 and 24 stand out.

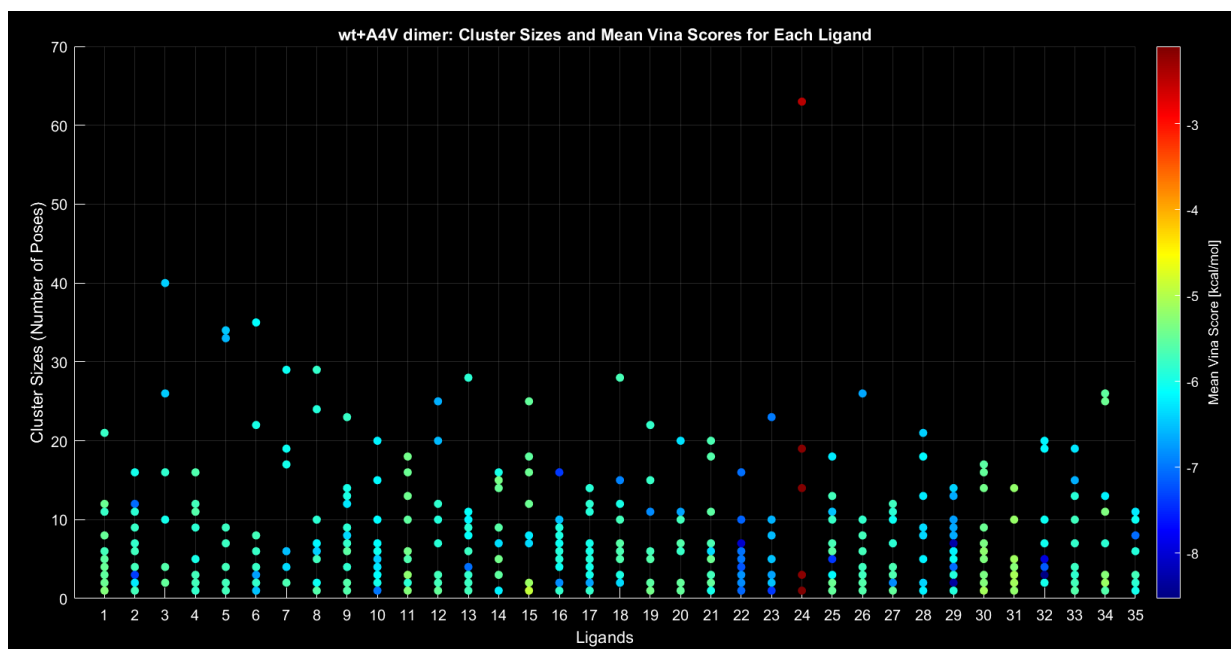


Figure 53: wt+A4V dimer clustering results

For the wt+G41D dimer (fig. 54), the ligands with high affinity are ligands 6, 7, 8, 9, 10, 17, 22, 23, 28, 29, 32, and 35, with ligands 22 and 23 standing out. Ligand 24 once again proves to have the worst affinity, although it is quite specific, reaching clusters of more than 30 elements. This also happens with ligand 23, which has both high affinity and a large cluster. The ligands with the largest clusters are 6 and 28, each with more than 30 elements.

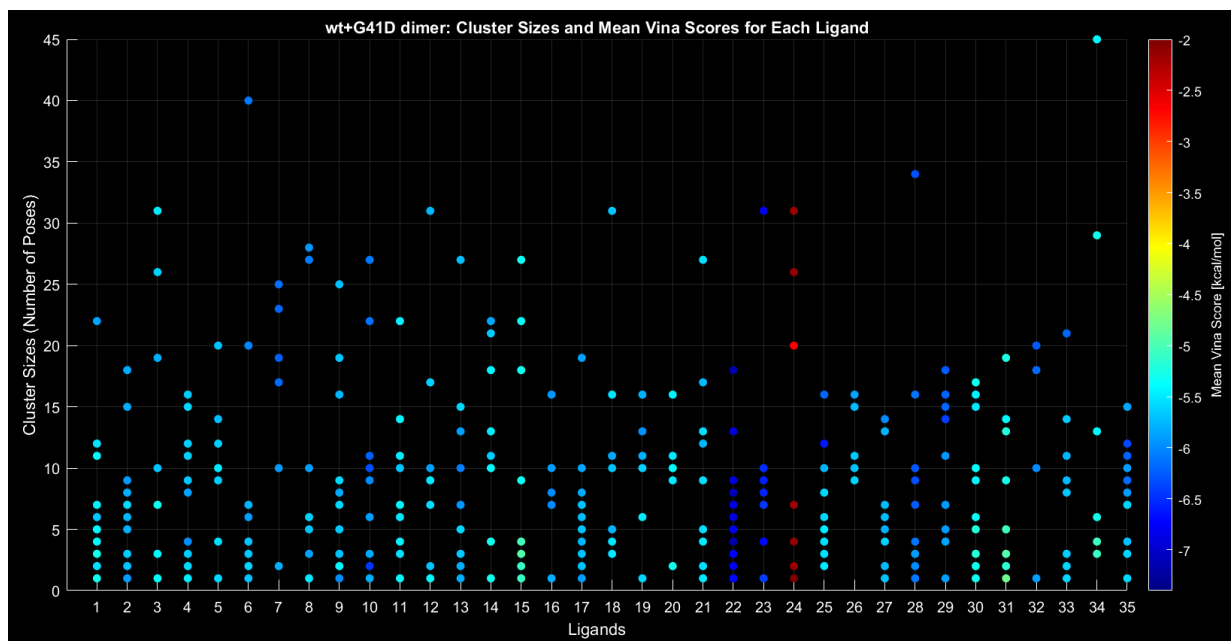


Figure 54: wt+G41D dimer clustering results

The trend for the wt+G41S dimer (fig. 55) is quite similar to that of wt+G41D. The cluster sizes generally stabilize at lower values, while the affinity values are quite similar to the previous case.



Figure 55: wt+G41S dimer clustering results

Regarding the wt+D76V dimer (fig. 56), the ligands with the highest affinity are 22 and 23, with ligand 29 also showing good affinity, albeit to a lesser extent. Ligand 24 continues to perform the worst, though it is quite specific. The ligand with the largest cluster is ligand 16, with a cluster of more than 50 poses.

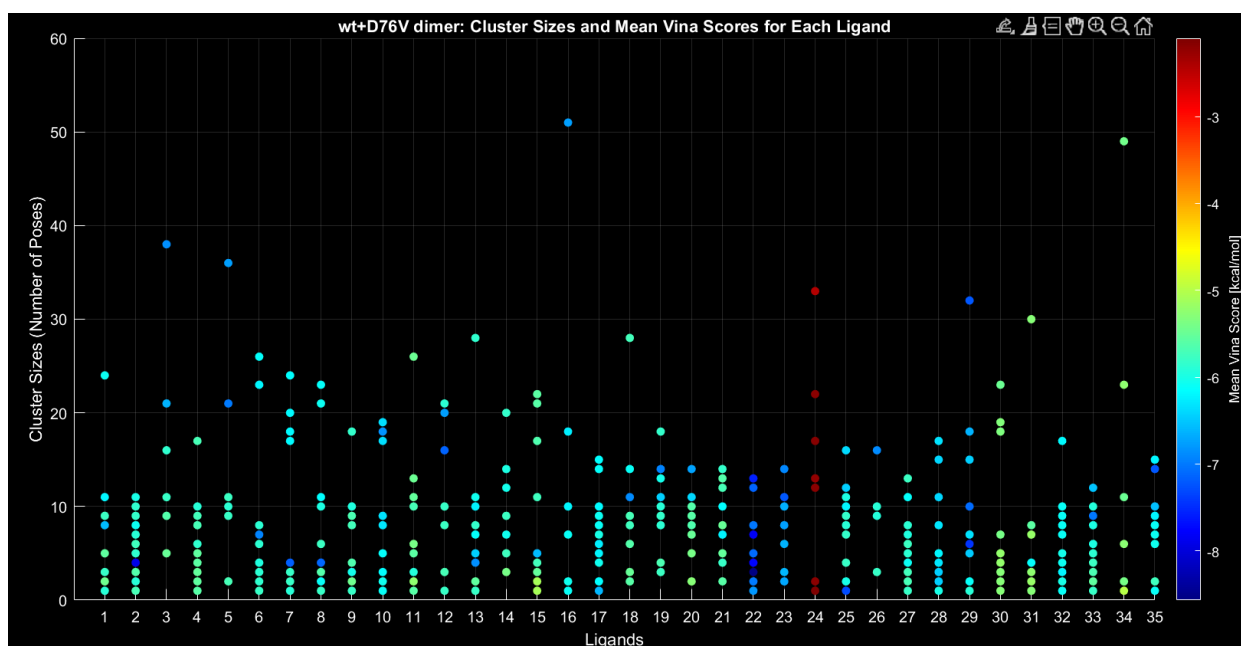


Figure 56: wt+D76V dimer clustering results

In the wt+G85R dimer (fig. 57), the previously observed trend continues, particularly with ligands 22 and 23 showing the highest affinity and ligand 24 the lowest.

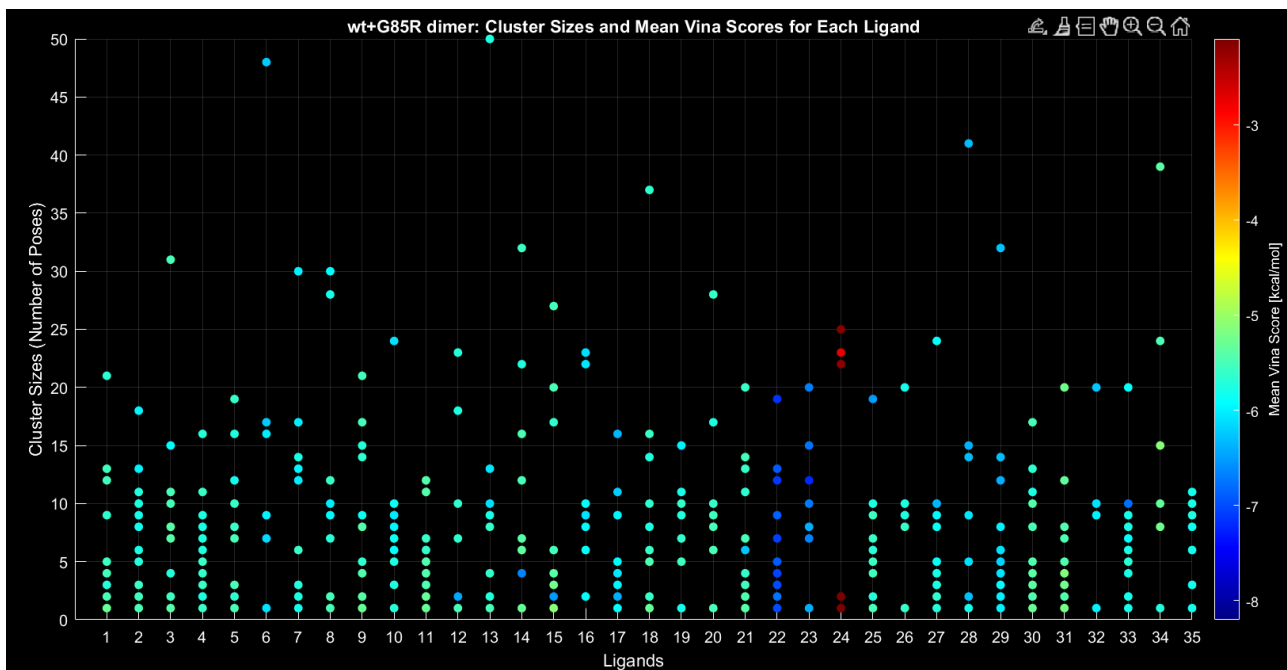


Figure 57: wt+G85R dimer clustering results

For the wt+G93A dimer (fig. 58), the ligands with the highest affinity are 16, 22, 23, 29, and 32, while those with greater specificity, indicated by the size of one of their clusters, are 3, 20, 25, and 24, the latter still showing poor affinity.

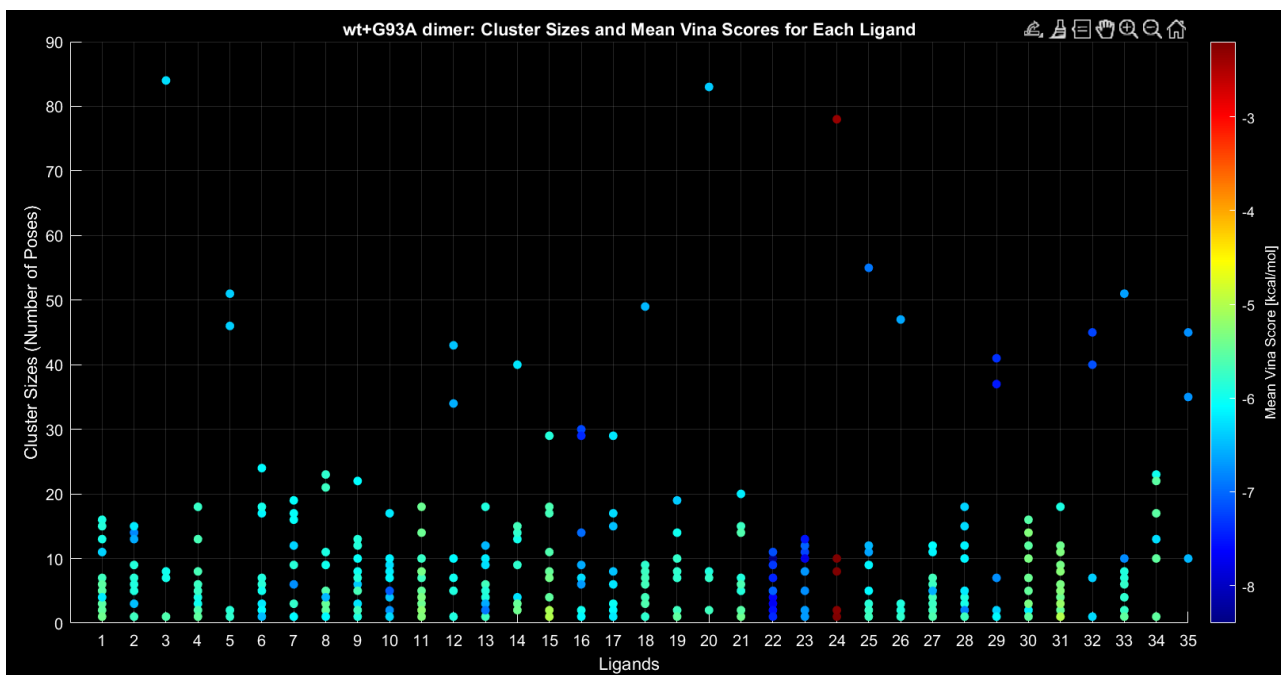


Figure 58: wt+G93A dimer clustering results

In the wt+I104F dimer (fig. 59), the general trend continues, with ligands 24 and 34 being among the most specific, and ligands 12, 16, 22, 23, 25, 29, and 35 having the highest affinity.

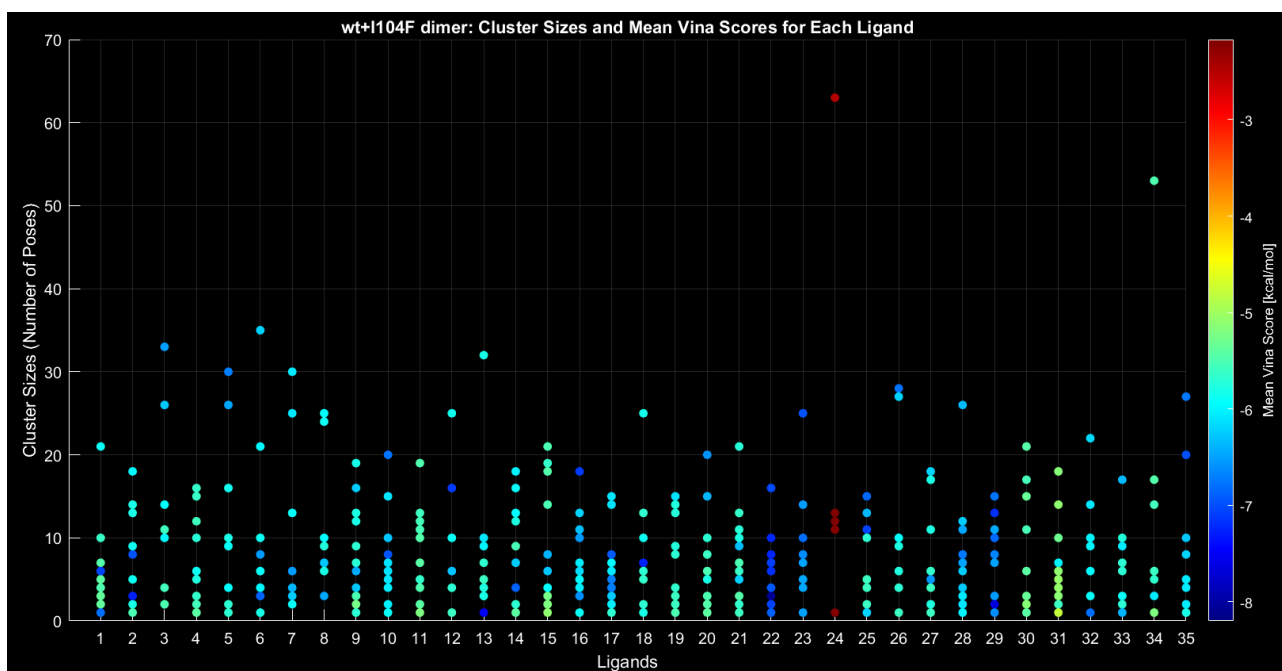


Figure 59: wt+I104F dimer clustering results

From the docking analysis of the different dimers, no significant differences emerge between the various mutations. The ligands with the highest affinity are generally the same across all dimers, with ligands 22 and 23, also 28 and 29, consistently standing out for their high affinity. At the same time, ligand 24, despite its low affinity, often forms very large clusters, suggesting a certain binding specificity, even though it is energetically unfavorable. Overall, the behavior of the ligands appears similar between the dimers, with no major variations due to the mutations.

Conclusion

The study aimed to identify compounds capable of inhibiting the infectious mechanism driven by SOD1 protein mutants, which could lead to either a loss of function of the protein or the acquisition of a toxic behavior. This mechanism is crucial for understanding the genetic causes of familial and sporadic ALS, making it a potential target for future therapies aimed at delaying the spread of this endogenous toxicity, mitigating the disease's effects, which is fatal within a few years, and thereby stopping the disease at its root. Initially, three-dimensional models were generated for seven mutated dimers and one wild-type dimer. Following 1200 ns of molecular dynamics simulations, data were collected, revealing some differences between the mutants and the wild type. Notable observations included greater fluctuations in the RMSD of the wt+A4V dimer compared to the other dimers, while other differences were less pronounced. Following the selection of an appropriate drug library, blind docking was performed between each compound and protein model, considering the entire protein without prior information on the potential binding site. It emerged that some drugs, such as Telbivudine, PRG-A01 and Thymidine Monophosphate appeared to have greater binding affinity than others, while S-XL6, already identified in the literature as an effective crosslinker of the protein's constituent monomers, despite showing the lowest binding affinity among the compounds, demonstrated the highest specificity, forming a sizable cluster of poses across various docking iterations and in nearly all dimers.

Future research

For future studies, it would be highly beneficial, depending on the available computational resources, to create a much larger pool of compounds and analyze a greater number of mutations, considering that over 200 mutations impact the SOD1 gene and protein. Furthermore, for the docking procedure, it would be worthwhile to increase the number of poses to obtain more consistent data regarding each ligand's binding region with the proteins, thereby identifying the binding sites. This approach could also involve applying molecular dynamics simulations to the dimer-ligand complexes to evaluate whether the presence of the ligand might have a beneficial effect, potentially influencing the protein's conformational variability and/or stability.

References

- [1] A. Eisen, S. Vucic e H. Mitsumoto, «History of ALS and the competing theories on pathogenesis: IFCN handbook chapter,» *Clinical Neurophysiology Practice*, vol. 9, pp. 2-4, 2023.
- [2] «Stanford Medicine,» [Online]. Available: <https://stanfordhealthcare.org/medical-conditions/brain-and-nerves/amyotrophic-lateral-sclerosis/types.html>.
- [3] S. Karceski, «Understanding the different types of ALS,» *Neurology*, vol. 94, n. 8, pp. e880-e883, 2020.
- [4] R. H. Shmerling, «Harvard Health Publishing,» [Online]. Available: <https://www.health.harvard.edu/blog/can-als-be-caused-by-traumatic-brain-injury-202202022680>.
- [5] D. H. Daneshvar, J. Mez, M. L. Alosco, Z. H. Baucom, I. Mahar, C. M. Baugh, J. P. Valle, J. Weuve, S. Paganoni, R. C. Cantu, R. D. Zafonte, R. A. Stern, T. D. Stein, Y. Tripodis e Nowinski, «Incidence of and Mortality From Amyotrophic Lateral Sclerosis in National Football League Athletes,» *JAMA Netw Open*, 2021.
- [6] S. Peters, A. E. Visser, F. D'Ovidio, E. Beghi, A. Chiò, G. Logroscino, O. Hardiman, H. Kromhout, A. Huss, J. Veldink, R. Vermeulen e L. H. van den Berg, «Associations of Electric Shock and Extremely Low-Frequency Magnetic Field Exposure With the Risk of Amyotrophic Lateral Sclerosis: The Euro-MOTOR Project,» *American Journal of Epidemiology*, vol. 188, p. 804, 2019.
- [7] A. Chiò, C. Moglia, A. Canosa, U. Manera, F. D'Ovidio, R. Vasta, M. Grassano, M. Brunetti, M. Barberis, L. Corrado, S. D'Alfonso, B. Iazzolino, L. Peotta e M. F. Sarnelli, «ALS phenotype is influenced by age, sex, and genetics: A population-based study,» *Neurology*, vol. 94, n. 8, pp. 802-810, 2020.
- [8] B. Swinnen e W. Robberecht, «The phenotypic variability of amyotrophic lateral sclerosis,» *Neurology*, pp. 1-3, 2014.
- [9] «alsnewstoday,» 2023. [Online]. Available: <https://alsnewstoday.com/forms-of-als/>.
- [10] «Protein,» [Online]. Available: <https://en.wikipedia.org/wiki/Protein>.
- [11] «medlineplus.gov,» [Online]. Available: <https://medlineplus.gov/genetics/understanding/howgeneswork/protein/>.
- [12] «National Human Genome Research Institute,» 2024. [Online]. Available: <https://www.genome.gov/genetics-glossary/Amino-Acids>.
- [13] T. E. Creighton, «Proteins: Secondary, Tertiary, and Quaternary Structure,» in *Protein Structure A Pratical Approach*, p. 159.
- [14] M. Kotb-El-Sayed, «Amino acids and protein chemistry part 1,» pp. 5-7, 2020.
- [15] T. E. Creighton, «Amino Acids,» in *Protein Structure A Pratical Approach*, pp. 1-6.
- [16] W. O. University, «CH451 Biochemistry II Review of Amino Acids and Chirality,» pp. 2-5.
- [17] M. Suzuki e J. Sasabe, «Distinctive Roles of D-Amino Acids in the Homochiral World: Chirality of Amino Acids Modulates Mammalian Physiology and Pathology,» *The Keio Journal of Medicine*, vol. 68, n. 1, 2018.
- [18] «University, Savitribai Phule Pune,» [Online]. Available: <https://soe.unipune.ac.in/studymaterial/swapnaGaikwadOnline/aminoacids-171113130407.pdf>.
- [19] «Polyprotic Acids & Bases,» 15 April 2024. [Online]. Available: <https://chem.libretexts.org/@go/page/55234>.

- [20] C. E. Barquilha e M. C. Braga, «Adsorption of organic and inorganic pollutants onto biochars: Challenges, operating conditions, and mechanisms,» *Bioresource Technology Reports*, 2021.
- [21] «Hydrophobic Interactions,» 2023. [Online]. Available: <https://chem.libretexts.org/@go/page/1506>.
- [22] «Van der Waals Forces,» 2023. [Online]. Available: <https://chem.libretexts.org/@go/page/1511>.
- [23] «Hydrogen Bonding,» 2023. [Online]. Available: <https://chem.libretexts.org/@go/page/1660>.
- [24] A. Fernández-Barbero, I. J. Suárez, B. Sierra-Martín, A. Fernández-Nieves, F. J. de las Nieves, M. Marquez, J. Rubio-Retama e E. López-Cabarcos, «Gels and microgels for nanotechnological applications,» *Advances in Colloid and Interface Science*, Vol. %1 di %2147-148, p. 90.
- [25] R. J. S., «BASIC ELEMENTS OF PROTEIN STRUCTURE,» in *The Anatomy and Taxonomy of Protein Structure*, 1981, pp. 181-182.
- [26] L. S e N. JS, «An artificial beta-sheet that dimerizes through parallel beta-sheet interactions.,» *Journal of the American Chemical Society*, vol. 129, n. 43, pp. 13043-13048, 2007.
- [27] «Beta hairpin,» [Online]. Available: https://en.wikipedia.org/wiki/Beta_hairpin.
- [28] N. Marco, «Struttura delle proteine,» 2010-2011.
- [29] K. Oberholser, M. Harel e I. Hanukoglu, «Proteopedia,» [Online]. Available: https://proteopedia.org/wiki/index.php/Fibrous_Proteins.
- [30] C.-H. Shen, *Diagnostic Molecular Biology*, 2019, pp. 249-276.
- [31] emach6, «Quizlet,» [Online]. Available: <https://quizlet.com/gb/232398154/biological-molecules-globular-and-fibrous-proteins-diagram/>.
- [32] D. Goodsell, «PDB-101,» 2007. [Online]. Available: <https://pdb101.rcsb.org/motm/94>.
- [33] «byjus,» 2021. [Online]. Available: <https://byjus.com/jee/disproportionation-reaction/>.
- [34] «Superoxide,» [Online]. Available: <https://en.wikipedia.org/wiki/Superoxide>.
- [35] M. Mraz, «Cu/Zn Superoxide dismutase,» 2016. [Online]. Available: https://chem.libretexts.org/Courses/Saint_Marys_College_Notre_Dame_IN/CHEM_342%3A_Bio-inorganic_Chemistry/Readings/Metals_in_Biological_Systems_%28Saint_Mary%27s_College%29/Antioxidant%3A_Cu_Zn_Superoxide_dismutase_%28SOD1%29.
- [36] M. Hayyan, M. A. Hashim e I. M. AlNashef, «Superoxide Ion: Generation and Chemical Implications,» *chemical review*, vol. 116, p. 3030, 2016.
- [37] «Hydroperoxyl,» [Online]. Available: <https://en.wikipedia.org/wiki/Hydroperoxyl>.
- [38] «PDB-101,» 2004. [Online]. Available: <https://pdb101.rcsb.org/motm/57>.
- [39] J. Ishrat e M. N. Shahid, «Conformational dynamics of superoxide dismutase (SOD1) in osmolytes: a molecular dynamics simulation study,» *The Royal Society of Chemistry*, vol. 10, pp. 27598-27614, 2020.
- [40] J. Perry, D. Shin, E. Getzoff e J. Tainer, «The structural biochemistry of the superoxide dismutases,» *Biochimica et Biophysica*, p. 246, 2010.
- [41] R. Byström, «SOD1's Law: An Investigation of ALS Provoking Properties in SOD1,» 2009.
- [42] F. Yang, *Biophysical chemistry of the ALS associated*, 2021.
- [43] R. Byström, «SOD1's Law: An Investigation of ALS Provoking Properties in SOD1,» p. 30.
- [44] F. Tafuri, D. Ronchi, F. Magri, G. P. Comi e S. Corti, «SOD1 misplacing and mitochondrial dysfunction in amyotrophic lateral sclerosis pathogenesis,» *Frontiers in cellular neuroscience*, 2015.

- [45] M. Berdyński, P. Miszta, K. Safranow, P. M. Andersen, M. Morita, S. Filipek, C. Żekanowski e M. Kuźma-Kozakiewicz, «SOD1 mutations associated with amyotrophic lateral sclerosis analysis of variant severity,» *Nature*, p. 1, 2022.
- [46] E. Bernard, A. Pegat, J. Svahn, F. Bouhour, P. Leblanc, S. Millecamps, S. Thobois, C. Guissart, S. Lumbroso e K. Mouzat, «Clinical and Molecular Landscape of ALS Patients with SOD1 Mutations: Novel Pathogenic Variants and Novel Phenotypes. A Single ALS Center Study,» *International Journal of Molecular Sciences*, 2020.
- [47] «alsod SOD1,» [Online]. Available: <https://alsod.ac.uk/output/gene.php/SOD1>.
- [48] L. J. Hayward, J. A. Rodriguez, J. W. Kim, A. Tiwari, J. J. Goto, D. E. Cabelli, J. Selverstone Valentine e R. H. Brown Jr, «Decreased metallation and activity in subsets of mutant superoxide dismutases associated with familial amyotrophic lateral sclerosis,» *The Journal of biology chemistry*, vol. 277, n. 18.
- [49] A. Tiwari e L. J. Hayward, «Familial amyotrophic lateral sclerosis mutants of copper/zinc superoxide dismutase are susceptible to disulfide reduction,» *Journal of Biological Chemistry*, vol. 278, n. 8, pp. 5984-5992, 2003.
- [50] A. G. Reaume, J. L. Elliott, E. K. Hoffman, N. W. Kowall, R. J. Ferrante, D. F. Siwek, H. M. Wilcox, D. G. Flood, M. F. Beal, R. H. Brown Jr, R. W. Scott e W. D. Snider, «Motor neurons in Cu/Zn superoxide dismutase-deficient mice develop normally but exhibit enhanced cell death after axonal injury,» *Nature genetics*, vol. 13, n. 1, pp. 43-7, 1996.
- [51] D. R. Borchelt, M. K. Lee, H. S. Slunt, M. Guarnieri, Z. S. Xu, P. C. Wong, R. H. Brown Jr, D. L. Price, S. S. Sisodia e D. W. Cleveland, «Superoxide dismutase 1 with mutations linked to familial amyotrophic lateral sclerosis possesses significant activity.,» *PNAS*, vol. 91, n. 17, p. 8292–8296, 1994.
- [52] A. G. Estévez, J. P. Crow, J. B. Sampson, C. Reiter, Y. Zhuang, G. J. Richardson, M. M. Tarpey, L. Barbeito e J. S. Beckman, «Induction of Nitric Oxide -- Dependent Apoptosis in Motor Neurons by Zinc-Deficient Superoxide Dismutase,» *Science*, vol. 286, n. 5449, pp. 2498-2500, 1979.
- [53] P. Pasinelli e R. H. Brown, «Molecular biology of amyotrophic lateral sclerosis: insights from genetics,» *Nature reviews, Neuroscience*, vol. 7, n. 9, pp. 710-723, 2006.
- [54] J. Selverstone Valentine, P. A. Doucette e S. Zittin Potter, «Copper-zinc superoxide dismutase and amyotrophic lateral sclerosis,» *Annual review of biochemistry*.
- [55] A. Nordlund e M. Oliveberg, «SOD1-associated ALS: a promising system for elucidating the origin of protein-misfolding disease,» *HFSP journal*, vol. 2, n. 6, pp. 354-364, 2008.
- [56] S. Chattopadhyay, A. Uysal, A. Uysal, B. Stripe e P. Dutta, vol. 107, n. 24.
- [57] D. Eisenberg e M. Jucker, «The amyloid state of proteins in human diseases,» *Cell*, vol. 148, n. 6, pp. 1188-203, 2012.
- [58] C. Münch, J. O'Brien e A. Bertolotti, «Prion-like propagation of mutant superoxide dismutase-1 misfolding in neuronal cells,» *PNAS*, vol. 108, n. 9, pp. 3548-53, 2011.
- [59] C. Münch e A. Bertolotti, «Exposure of hydrophobic surfaces initiates aggregation of diverse ALS-causing superoxide dismutase-1 mutants,» *Journal of molecular biology*, pp. 512-25, 2010.
- [60] S. B. Prusiner, «Prions,» *PNAS*, vol. 95, pp. 13363-13383, 1998, .
- [61] G. C. William, «Template-Directed Protein Misfolding in Neurodegenerative Disease,» 2012.
- [62] A. T. W, D. G. Michèle, N.-P. Kim e L. A. L. Patricia, «Reduced Abundance and Subverted Functions of Proteins in Prion-Like Diseases: Gained Functions Fascinate but Lost Functions Affect Aetiology,» *International journal of molecular sciences*, 2017.

- [63] M. Rodrigo, M.-G. Ines e S. Claudio, «Cross-Seeding of Misfolded Proteins: Implications for Etiology and Pathogenesis of Protein Misfolding Diseases,» *PLOS Pathogens*, 2013.
- [64] W. J. S. e H. A. L., «Deadly Conformations—Protein Misfolding in Prion Disease,» *Cell*, vol. 89, n. 4, p. 499–510, 1997.
- [65] M. Mirdita, K. Schütze, Y. Moriwaki, L. Heo, S. Ovchinnikov e M. Steinegger, «ColabFold v1.5.5: AlphaFold2 using MMseqs2,» [Online]. Available: <https://colab.research.google.com/github/sokrypton/ColabFold/blob/main/AlphaFold2.ipynb>.
- [66] Bovinlab, «HADDOCK 2.4,» [Online]. Available: <https://rascar.science.uu.nl/haddock2.4/>.
- [67] R. Honorato, M. Trellet, B. Jiménez-García, J. Schaarschmidt, M. Giulini, V. Reys, P. Koukos, J. Rodrigues, E. Karaca, G. van Zundert, J. Roel-Touris, C. van Noort, Z. Jandová, A. Melquiond e A. Bonvin, «The HADDOCK2.4 web server: A leap forward in integrative modelling of biomolecular complexes,» *nature protocols*, 2024.
- [68] R. Honorato, P. Koukos, B. Jimenez-Garcia, A. Tsaregorodtsev, M. Verlatto, A. Giachetti, A. Rosato e A. Bonvin, «Structural biology in the clouds: The WeNMR-EOSC Ecosystem,» *frontiers*, vol. 8, 2021.
- [69] [Online]. Available: <https://wenmr.science.uu.nl/pdbtools/>.
- [70] M. M. R. e P. Sergio, «Split the Charge Difference in Two! A Rule of Thumb for Adding Proper Amounts of Ions in MD Simulations,» *Journal of Chemical Theory and Computaiton*, pp. 1367-1372, 2020.
- [71] T.-G. Woo, M.-H. Yoon, S.-m. Kang, S. Park, J.-H. Cho, Y. J. Hwang, J. Ahn, H. Jang, Y.-J. Shin, E.-M. Jung, N.-C. Ha, B.-H. Kim, Y. Kwon e B.-J. Park, «Novel chemical inhibitor against SOD1 misfolding and aggregation protects neuron-loss and ameliorates disease symptoms in ALS mouse model,» *Communications biology*, pp. 1-12, 2021.
- [72] «ProbeChem,» [Online]. Available: https://www.probechem.com/products_PRG-A01.html.
- [73] «Chemical Sketch Tool,» [Online]. Available: <https://www.rcsb.org/chemical-sketch>.
- [74] M. G. DuVal, V. K. Hinge, N. Snyder, R. Kanyo, J. Bratvold, E. Pokrishevsky, N. R. Cashman, N. Blinov, A. Kovalenko e W. T. Allison, «Tryptophan 32 mediates SOD1 toxicity in a in vivo motor neuron model of ALS and is a promising target for small molecule therapeutics,» *Neurobiology of disease*, 2018 .
- [75] M. A. Hossain, R. Sarin, D. P. Donnelly, B. C. Miller, A. Weiss, L. McAlary, S. V. Antonyuk, J. P. Salisbury, J. Amin, J. B. Conway, S. S. Watson, J. N. Winters, Y. Xu, N. Alam e Brahme, «Evaluating protein cross-linking as a therapeutic strategy to stabilize SOD1 variants in a mouse model of familial ALS,» *Plos biology*, 2024.
- [76] R. C. Gupta, I. R. M. Mukherjee, J. K. Malik, R. B. Doss, W.-D. Dettbarn e D. Milatovic, «Insecticides,» in *Biomarkers in Toxicology* , Ramesh C. Gupta, 2019, pp. 455-475.
- [77] F. Kamela, D. M. Umbach, R. S. Bedlack, M. Richards, M. Watson, M. C. Alavanja, A. Blair, J. A. Hoppin, S. Schmidt e D. P. Sandler, «PESTICIDE EXPOSURE AND AMYOTROPHIC LATERAL,» *Neurotoxicology*, vol. 33, n. 3, pp. 457-462, 2012.
- [78] S. Shabalala, C. Muller, J. Louw e R. Johnson, «Polyphenols, autophagy and doxorubicin-induced cardiotoxicity,» *Life Sciences*, vol. 180, pp. 160-170, 2017.
- [79] «Epigallocatechin Gallate,» [Online]. Available: <https://pubchem.ncbi.nlm.nih.gov/compound/Epigallocatechin-Gallate>.
- [80] K. S. P. e K. P., «Interaction between dimer interface residues of native and mutated SOD1 protein: a theoretical study,» *J Biol Inorg Chem*, pp. 509-510, 2015.

Appendix 1: Software

- **MATLAB R2023b**: MathWorks. MATLAB Version R2023b. The MathWorks Inc., Natick, Massachusetts, 2023. Available at: <https://mathworks.com>.
- **Amber 22**: Case, D.A., et al. AMBER 22. University of California, San Francisco, 2022. Available at: <https://ambermd.org>.
 - 1) D.A. Case, H.M. Aktulga, K. Belfon, I.Y. Ben-Shalom, J.T. Berryman, S.R. Brozell, D.S. Cerutti, T.E. Cheatham, III, G.A. Cisneros, V.W.D. Cruzeiro, T.A. Darden, N. Forouzes, M. Ghazimirsaeed, G. Giambaşu, T. Giese, M.K. Gilson, H. Gohlke, A.W. Goetz, J. Harris, Z. Huang, S. Izadi, S.A. Izmailov, K. Kasavajhala, M.C. Kaymak, A. Kovalenko, T. Kurtzman, T.S. Lee, P. Li, Z. Li, C. Lin, J. Liu, T. Luchko, R. Luo, M. Machado, M. Manathunga, K.M. Merz, Y. Miao, O. Mikhailovskii, G. Monard, H. Nguyen, K.A. O'Hearn, A. Onufriev, F. Pan, S. Pantano, A. Rahnamoun, D.R. Roe, A. Roitberg, C. Sagui, S. Schott-Verdugo, A. Shajan, J. Shen, C.L. Simmerling, N.R. Skrynnikov, J. Smith, J. Swails, R.C. Walker, J. Wang, J. Wang, X. Wu, Y. Wu, Y. Xiong, Y. Xue, D.M. York, C. Zhao, Q. Zhu, and P.A. Kollman (2024), Amber 2024, University of California, San Francisco.
 - 2) D.A. Case, H.M. Aktulga, K. Belfon, D.S. Cerutti, G.A. Cisneros, V.W.D. Cruz eiro, N. Forouzes, T.J. Giese, A.W. Götz, H. Gohlke, S. Izadi, K. Kasavajhala, M.C. Kaymak, E. King, T. Kurtzman, T.-S. Lee, P. Li, J. Liu, T. Luchko, R. Luo, M. Manathunga, M.R. Machado, H.M. Nguyen, K.A. O'Hearn, A.V. Onufriev, F. Pan, S. Pantano, R. Qi, A. Rahnamoun, A. Risheh, S. Schott-Verdugo, A. Shajan, J. Swails, J. Wang, H. Wei, X. Wu, Y. Wu, S. Zhang, S. Zhao, Q. Zhu, T.E. Cheatham III, D.R. Roe, A. Roitberg, C. Simmerling, D.M. York, M.C. Nagan*, and K.M. Merz Jr.* AmberTools. *J. Chem. Inf. Model.* **63**, 6183-6191 (2023).
- **Python 3**: Python Software Foundation. Python Language Reference, Version 3.x. Available at: <https://www.python.org>.
 - Van Rossum, G., & Drake, F. L. (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.
- **HADDOCK**: van Zundert, G.C.P., et al. HADDOCK2.2: a versatile protein-protein docking software. *Journal of Molecular Biology*, 428(4), 720-725, 2016. Available at: <https://wenmr.science.uu.nl/haddock2.4>.
 - Cyril Dominguez, Rolf Boelens and Alexandre M.J.J. Bonvin. [HADDOCK: a protein-protein docking approach based on biochemical and/or biophysical information.](#) *J. Am. Chem. Soc.* **125**, 1731-1737 (2003).
 - G.C.P van Zundert, J.P.G.L.M. Rodrigues, M. Trellet, C. Schmitz, P.L. Kastritis, E. Karaca, A.S.J. Melquiond, M. van Dijk, S.J. de Vries and A.M.J.J. Bonvin. "[The HADDOCK2.2 webserver: User-friendly integrative modeling of biomolecular complexes.](#)" *J. Mol. Biol.*, **428**, 720-725 (2016).
- **MOE (Molecular Operating Environment)**: Molecular Operating Environment (MOE), Version 2022.22 Chemical Computing Group Inc., Montreal, Canada. Available at: <https://www.chemcomp.com>.

- **AutoDock Tools 1.5.7:** Morris, G.M., et al. AutoDock Tools Version 1.5.7. Scripps Research Institute, 2009. Available at: <https://autodock.scripps.edu>.
- **AutoDock Vina:** Trott, O., and Olson, A.J. AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. Journal of Computational Chemistry, 31(2), 455-461, 2010. Available at: <https://vina.scripps.edu>.

Appendix 2: Scripts

MATLAB script 1: fasta mutated sequence conversion

```
%Matlab code
clc
clear
close

%Choose WT type .fasta file
[file1, path1] = uigetfile('*.fasta');
[~,Protein_template] = fastaread([path1,file1]);

%Choose mutations list .txt file
[file2, path2] = uigetfile('*.txt');
Protein_mut_table = readtable([path2,file2]);

L = length(Protein_template); %sequence length
N = height(Protein_mut_table); %number of mutations
i = 1;

letters_pre_mut = Protein_mut_table.Var1;
letters_pre_mut = char(letters_pre_mut);
numbers_mut = Protein_mut_table.Var2;
letters_mut = Protein_mut_table.Var3;
letters_mut = char(letters_mut);

Protein_support_matrix = Protein_template;
for i = 1 : N-1
    Protein_support_matrix = [Protein_support_matrix;Protein_template];
end

Protein_mut_matrix = Protein_support_matrix;
for i = 1 : N
    Protein_mut_matrix(i,numbers_mut(i)) = letters_mut(i);
end

% for i = 1 : N
%     data(i).Sequence = Protein_mut_matrix(i,:);
%     data(i).Header = [num2str(i) 'Mutation'];
% end

for i = 1 : N
    data(i).Sequence = Protein_mut_matrix(i,:);
    nn = numbers_mut(i);
    data(i).Header = ['Sequence: ' num2str(i) ', Mutation identifier: ' letters_pre_mut(i)
num2str(nn) letters_mut(i)  '];
end

delete mutated_sequences.fasta;

fastawrite('mutated_sequences.fasta', data)
```

Python script 1: changing atom and residue type of Zn-Cu ions

```
import tkinter as tk
from tkinter import filedialog
from biopandas.pdb import PandasPdb

def rename_atoms_and_residues(ppdb):
    # Modify atom and residue types for 'ATOM'
    for idx, row in ppdb.df['ATOM'].iterrows():
        if row['atom_name'].strip() == 'ZN+2':
            ppdb.df['ATOM'].at[idx, 'atom_name'] = 'ZN'
        if row['residue_name'].strip() == 'ZN2':
            ppdb.df['ATOM'].at[idx, 'residue_name'] = 'ZN'
        if row['atom_name'].strip() == 'CU+2':
            ppdb.df['ATOM'].at[idx, 'atom_name'] = 'CU'
        if row['residue_name'].strip() == 'CU2':
            ppdb.df['ATOM'].at[idx, 'residue_name'] = 'CU'

    # Modify atom and residue types for 'HETATM'
    for idx, row in ppdb.df['HETATM'].iterrows():
        if row['atom_name'].strip() == 'ZN+2':
            ppdb.df['HETATM'].at[idx, 'atom_name'] = 'ZN'
        if row['residue_name'].strip() == 'ZN2':
            ppdb.df['HETATM'].at[idx, 'residue_name'] = 'ZN'
        if row['atom_name'].strip() == 'CU+2':
            ppdb.df['HETATM'].at[idx, 'atom_name'] = 'CU'
        if row['residue_name'].strip() == 'CU2':
            ppdb.df['HETATM'].at[idx, 'residue_name'] = 'CU'

    return ppdb

def main():
    # Create a dialog window
    root = tk.Tk()
    root.withdraw() # Hide the main window

    # Open a dialog to select the PDB file
    file_path = filedialog.askopenfilename(title="Select a PDB file",
                                           filetype= (("PDB files", "*.pdb"), ("All
files", "*.*")))
    if not file_path:
        print("No file selected.")
        return

    # Load the PDB file
    ppdb = PandasPdb().read_pdb(file_path)

    # Rename atoms and residues
    ppdb = rename_atoms_and_residues(ppdb)

    # Open a dialog to select the save path
    save_path = filedialog.asksaveasfilename(defaultextension=".pdb",
                                             filetype= (("PDB files", "*.pdb"), ("All
files", "*.*")),
                                             title="Save the renamed file as")
    if not save_path:
        print("No save path selected.")
        return

    # Save the new PDB file
    ppdb.to_pdb(path=save_path, records=['ATOM', 'HETATM'], gz=False)

    print(f"Renamed file saved as: {save_path}")

if __name__ == "__main__":
    main()
```

MATLAB script 2: calculation of the required Na⁺ and Cl⁻ ions for charge neutralization

```
%input
Q = -4;
Nw = 12784;

%computing
Co = 0.15;
No = ceil(Nw*Co/56);

I = abs(No/Q);

if I >= 1
    r = mod(Q,2);
    if r == 0
        Nplus = (No - Q/2);
        Nminus = (No + Q/2);
    else
        Nplus = ceil(No - Q/2);
        Nminus = ceil(No + Q/2);
    end
end
end
```

CPPTRAJ script 1: RMSD, RMSF and radius of gyration calculation

```
parm monomer_amber_solv_1264.prmtop      # Load the topology file for
molecular simulation.
trajin md_trajectories.nc                # Load the trajectory file containing
the simulation frames.

strip :WAT,Cl-,Na+                       # Remove water molecules (WAT) and
ions (Cl-, Na+) from the trajectory.

# RMSD calculation by aligning the trajectory frames to the first frame using
backbone atoms
rms first @CA,C,N,O out md_rmsd_backbone.txt

# Alignment and calculation of Root Mean Square Fluctuations (RMSF) for backbone
atoms
rms first @CA,C,N,O
atomicfluct out md_rmsf_backbone.txt @CA,C,N,O byres

# Calculation of the radius of gyration for the protein backbone only
radgyr @CA,C,N,O out md_radgyr_backbone.txt

run                                       # Execute all the previous commands.
```

CPPTRAJ script 2: SASA calculation for residue

```
# Load modules and trajectories
parm dimer_c_amber_solv_1264_wt_I104F.prmtop # Load the topology file
trajin Sim1_run_wt_I104F_md_trajectories.nc # Load the trajectory file

# SASA measurement integrated over all residues
surf :1-306 out sasa.txt

# First group (1-19 residues)
surf :1 out sasa_per_residue_matrix1.txt
surf :2 out sasa_per_residue_matrix1.txt
surf :3 out sasa_per_residue_matrix1.txt
surf :4 out sasa_per_residue_matrix1.txt
surf :5 out sasa_per_residue_matrix1.txt
surf :6 out sasa_per_residue_matrix1.txt
surf :7 out sasa_per_residue_matrix1.txt
surf :8 out sasa_per_residue_matrix1.txt
surf :9 out sasa_per_residue_matrix1.txt
surf :10 out sasa_per_residue_matrix1.txt
surf :11 out sasa_per_residue_matrix1.txt
surf :12 out sasa_per_residue_matrix1.txt
surf :13 out sasa_per_residue_matrix1.txt
surf :14 out sasa_per_residue_matrix1.txt
surf :15 out sasa_per_residue_matrix1.txt
surf :16 out sasa_per_residue_matrix1.txt
surf :17 out sasa_per_residue_matrix1.txt
surf :18 out sasa_per_residue_matrix1.txt
surf :19 out sasa_per_residue_matrix1.txt

# Second group (20-38 residues)
surf :20 out sasa_per_residue_matrix2.txt
surf :21 out sasa_per_residue_matrix2.txt
surf :22 out sasa_per_residue_matrix2.txt
surf :23 out sasa_per_residue_matrix2.txt
surf :24 out sasa_per_residue_matrix2.txt
surf :25 out sasa_per_residue_matrix2.txt
surf :26 out sasa_per_residue_matrix2.txt
surf :27 out sasa_per_residue_matrix2.txt
surf :28 out sasa_per_residue_matrix2.txt
surf :29 out sasa_per_residue_matrix2.txt
surf :30 out sasa_per_residue_matrix2.txt
surf :31 out sasa_per_residue_matrix2.txt
surf :32 out sasa_per_residue_matrix2.txt
surf :33 out sasa_per_residue_matrix2.txt
surf :34 out sasa_per_residue_matrix2.txt
surf :35 out sasa_per_residue_matrix2.txt
surf :36 out sasa_per_residue_matrix2.txt
surf :37 out sasa_per_residue_matrix2.txt
surf :38 out sasa_per_residue_matrix2.txt

# Third group (39-57 residues)
surf :39 out sasa_per_residue_matrix3.txt
surf :40 out sasa_per_residue_matrix3.txt
surf :41 out sasa_per_residue_matrix3.txt
surf :42 out sasa_per_residue_matrix3.txt
surf :43 out sasa_per_residue_matrix3.txt
surf :44 out sasa_per_residue_matrix3.txt
surf :45 out sasa_per_residue_matrix3.txt
surf :46 out sasa_per_residue_matrix3.txt
surf :47 out sasa_per_residue_matrix3.txt
surf :48 out sasa_per_residue_matrix3.txt
```



```
surf :49 out sasa_per_residue_matrix3.txt
surf :50 out sasa_per_residue_matrix3.txt
surf :51 out sasa_per_residue_matrix3.txt
surf :52 out sasa_per_residue_matrix3.txt
surf :53 out sasa_per_residue_matrix3.txt
surf :54 out sasa_per_residue_matrix3.txt
surf :55 out sasa_per_residue_matrix3.txt
surf :56 out sasa_per_residue_matrix3.txt
surf :57 out sasa_per_residue_matrix3.txt
```

Fourth group (58-76 residues)

```
surf :58 out sasa_per_residue_matrix4.txt
surf :59 out sasa_per_residue_matrix4.txt
surf :60 out sasa_per_residue_matrix4.txt
surf :61 out sasa_per_residue_matrix4.txt
surf :62 out sasa_per_residue_matrix4.txt
surf :63 out sasa_per_residue_matrix4.txt
surf :64 out sasa_per_residue_matrix4.txt
surf :65 out sasa_per_residue_matrix4.txt
surf :66 out sasa_per_residue_matrix4.txt
surf :67 out sasa_per_residue_matrix4.txt
surf :68 out sasa_per_residue_matrix4.txt
surf :69 out sasa_per_residue_matrix4.txt
surf :70 out sasa_per_residue_matrix4.txt
surf :71 out sasa_per_residue_matrix4.txt
surf :72 out sasa_per_residue_matrix4.txt
surf :73 out sasa_per_residue_matrix4.txt
surf :74 out sasa_per_residue_matrix4.txt
surf :75 out sasa_per_residue_matrix4.txt
surf :76 out sasa_per_residue_matrix4.txt
```

Fifth group (77-95 residues)

```
surf :77 out sasa_per_residue_matrix5.txt
surf :78 out sasa_per_residue_matrix5.txt
surf :79 out sasa_per_residue_matrix5.txt
surf :80 out sasa_per_residue_matrix5.txt
surf :81 out sasa_per_residue_matrix5.txt
surf :82 out sasa_per_residue_matrix5.txt
surf :83 out sasa_per_residue_matrix5.txt
surf :84 out sasa_per_residue_matrix5.txt
surf :85 out sasa_per_residue_matrix5.txt
surf :86 out sasa_per_residue_matrix5.txt
surf :87 out sasa_per_residue_matrix5.txt
surf :88 out sasa_per_residue_matrix5.txt
surf :89 out sasa_per_residue_matrix5.txt
surf :90 out sasa_per_residue_matrix5.txt
surf :91 out sasa_per_residue_matrix5.txt
surf :92 out sasa_per_residue_matrix5.txt
surf :93 out sasa_per_residue_matrix5.txt
surf :94 out sasa_per_residue_matrix5.txt
surf :95 out sasa_per_residue_matrix5.txt
```

Sixth group (96-114 residues)

```
surf :96 out sasa_per_residue_matrix6.txt
surf :97 out sasa_per_residue_matrix6.txt
surf :98 out sasa_per_residue_matrix6.txt
surf :99 out sasa_per_residue_matrix6.txt
surf :100 out sasa_per_residue_matrix6.txt
surf :101 out sasa_per_residue_matrix6.txt
surf :102 out sasa_per_residue_matrix6.txt
surf :103 out sasa_per_residue_matrix6.txt
surf :104 out sasa_per_residue_matrix6.txt
```

```
surf :105 out sasa_per_residue_matrix6.txt
surf :106 out sasa_per_residue_matrix6.txt
surf :107 out sasa_per_residue_matrix6.txt
surf :108 out sasa_per_residue_matrix6.txt
surf :109 out sasa_per_residue_matrix6.txt
surf :110 out sasa_per_residue_matrix6.txt
surf :111 out sasa_per_residue_matrix6.txt
surf :112 out sasa_per_residue_matrix6.txt
surf :113 out sasa_per_residue_matrix6.txt
surf :114 out sasa_per_residue_matrix6.txt
```

Seventh group (115-133 residues)

```
surf :115 out sasa_per_residue_matrix7.txt
surf :116 out sasa_per_residue_matrix7.txt
surf :117 out sasa_per_residue_matrix7.txt
surf :118 out sasa_per_residue_matrix7.txt
surf :119 out sasa_per_residue_matrix7.txt
surf :120 out sasa_per_residue_matrix7.txt
surf :121 out sasa_per_residue_matrix7.txt
surf :122 out sasa_per_residue_matrix7.txt
surf :123 out sasa_per_residue_matrix7.txt
surf :124 out sasa_per_residue_matrix7.txt
surf :125 out sasa_per_residue_matrix7.txt
surf :126 out sasa_per_residue_matrix7.txt
surf :127 out sasa_per_residue_matrix7.txt
surf :128 out sasa_per_residue_matrix7.txt
surf :129 out sasa_per_residue_matrix7.txt
surf :130 out sasa_per_residue_matrix7.txt
surf :131 out sasa_per_residue_matrix7.txt
surf :132 out sasa_per_residue_matrix7.txt
surf :133 out sasa_per_residue_matrix7.txt
```

Eighth group (134-152 residues)

```
surf :134 out sasa_per_residue_matrix8.txt
surf :135 out sasa_per_residue_matrix8.txt
surf :136 out sasa_per_residue_matrix8.txt
surf :137 out sasa_per_residue_matrix8.txt
surf :138 out sasa_per_residue_matrix8.txt
surf :139 out sasa_per_residue_matrix8.txt
surf :140 out sasa_per_residue_matrix8.txt
surf :141 out sasa_per_residue_matrix8.txt
surf :142 out sasa_per_residue_matrix8.txt
surf :143 out sasa_per_residue_matrix8.txt
surf :144 out sasa_per_residue_matrix8.txt
surf :145 out sasa_per_residue_matrix8.txt
surf :146 out sasa_per_residue_matrix8.txt
surf :147 out sasa_per_residue_matrix8.txt
surf :148 out sasa_per_residue_matrix8.txt
surf :149 out sasa_per_residue_matrix8.txt
surf :150 out sasa_per_residue_matrix8.txt
surf :151 out sasa_per_residue_matrix8.txt
surf :152 out sasa_per_residue_matrix8.txt
```

Ninth group (153-171 residues)

```
surf :153 out sasa_per_residue_matrix9.txt
surf :154 out sasa_per_residue_matrix9.txt
surf :155 out sasa_per_residue_matrix9.txt
surf :156 out sasa_per_residue_matrix9.txt
surf :157 out sasa_per_residue_matrix9.txt
surf :158 out sasa_per_residue_matrix9.txt
surf :159 out sasa_per_residue_matrix9.txt
surf :160 out sasa_per_residue_matrix9.txt
```

```
surf :161 out sasa_per_residue_matrix9.txt
surf :162 out sasa_per_residue_matrix9.txt
surf :163 out sasa_per_residue_matrix9.txt
surf :164 out sasa_per_residue_matrix9.txt
surf :165 out sasa_per_residue_matrix9.txt
surf :166 out sasa_per_residue_matrix9.txt
surf :167 out sasa_per_residue_matrix9.txt
surf :168 out sasa_per_residue_matrix9.txt
surf :169 out sasa_per_residue_matrix9.txt
surf :170 out sasa_per_residue_matrix9.txt
surf :171 out sasa_per_residue_matrix9.txt
```

Tenth group (172-190 residues)

```
surf :172 out sasa_per_residue_matrix10.txt
surf :173 out sasa_per_residue_matrix10.txt
surf :174 out sasa_per_residue_matrix10.txt
surf :175 out sasa_per_residue_matrix10.txt
surf :176 out sasa_per_residue_matrix10.txt
surf :177 out sasa_per_residue_matrix10.txt
surf :178 out sasa_per_residue_matrix10.txt
surf :179 out sasa_per_residue_matrix10.txt
surf :180 out sasa_per_residue_matrix10.txt
surf :181 out sasa_per_residue_matrix10.txt
surf :182 out sasa_per_residue_matrix10.txt
surf :183 out sasa_per_residue_matrix10.txt
surf :184 out sasa_per_residue_matrix10.txt
surf :185 out sasa_per_residue_matrix10.txt
surf :186 out sasa_per_residue_matrix10.txt
surf :187 out sasa_per_residue_matrix10.txt
surf :188 out sasa_per_residue_matrix10.txt
surf :189 out sasa_per_residue_matrix10.txt
surf :190 out sasa_per_residue_matrix10.txt
```

11th group (191-209 residues)

```
surf :191 out sasa_per_residue_matrix11.txt
surf :192 out sasa_per_residue_matrix11.txt
surf :193 out sasa_per_residue_matrix11.txt
surf :194 out sasa_per_residue_matrix11.txt
surf :195 out sasa_per_residue_matrix11.txt
surf :196 out sasa_per_residue_matrix11.txt
surf :197 out sasa_per_residue_matrix11.txt
surf :198 out sasa_per_residue_matrix11.txt
surf :199 out sasa_per_residue_matrix11.txt
surf :200 out sasa_per_residue_matrix11.txt
surf :201 out sasa_per_residue_matrix11.txt
surf :202 out sasa_per_residue_matrix11.txt
surf :203 out sasa_per_residue_matrix11.txt
surf :204 out sasa_per_residue_matrix11.txt
surf :205 out sasa_per_residue_matrix11.txt
surf :206 out sasa_per_residue_matrix11.txt
surf :207 out sasa_per_residue_matrix11.txt
surf :208 out sasa_per_residue_matrix11.txt
surf :209 out sasa_per_residue_matrix11.txt
```

12th group (210-228 residues)

```
surf :210 out sasa_per_residue_matrix12.txt
surf :211 out sasa_per_residue_matrix12.txt
surf :212 out sasa_per_residue_matrix12.txt
surf :213 out sasa_per_residue_matrix12.txt
surf :214 out sasa_per_residue_matrix12.txt
surf :215 out sasa_per_residue_matrix12.txt
surf :216 out sasa_per_residue_matrix12.txt
```



```
surf :273 out sasa_per_residue_matrix15.txt
surf :274 out sasa_per_residue_matrix15.txt
surf :275 out sasa_per_residue_matrix15.txt
surf :276 out sasa_per_residue_matrix15.txt
surf :277 out sasa_per_residue_matrix15.txt
surf :278 out sasa_per_residue_matrix15.txt
surf :279 out sasa_per_residue_matrix15.txt
surf :280 out sasa_per_residue_matrix15.txt
surf :281 out sasa_per_residue_matrix15.txt
surf :282 out sasa_per_residue_matrix15.txt
surf :283 out sasa_per_residue_matrix15.txt
surf :284 out sasa_per_residue_matrix15.txt
surf :285 out sasa_per_residue_matrix15.txt
```

```
surf :286 out sasa_per_residue_matrix16.txt
surf :287 out sasa_per_residue_matrix16.txt
surf :288 out sasa_per_residue_matrix16.txt
surf :289 out sasa_per_residue_matrix16.txt
surf :290 out sasa_per_residue_matrix16.txt
surf :291 out sasa_per_residue_matrix16.txt
surf :292 out sasa_per_residue_matrix16.txt
surf :293 out sasa_per_residue_matrix16.txt
surf :294 out sasa_per_residue_matrix16.txt
surf :295 out sasa_per_residue_matrix16.txt
surf :296 out sasa_per_residue_matrix16.txt
surf :297 out sasa_per_residue_matrix16.txt
surf :298 out sasa_per_residue_matrix16.txt
surf :299 out sasa_per_residue_matrix16.txt
surf :300 out sasa_per_residue_matrix16.txt
surf :301 out sasa_per_residue_matrix16.txt
surf :302 out sasa_per_residue_matrix16.txt
surf :303 out sasa_per_residue_matrix16.txt
surf :304 out sasa_per_residue_matrix16.txt
```

```
# 17th group (305-306 residues)
```

```
surf :305 out sasa_per_residue_matrix17.txt
surf :306 out sasa_per_residue_matrix17.txt
```

CPPTRAJ script 3: conformational clustering

```
# Load the original topology file with the second monomer mutated to I104F
parm dimer_c_amber_solv_1264_wt_I104F.prmtop

trajin Sim1_run_wt_I104F_md_trajectories.nc

# Recenter the molecules
autoimage

# Remove water molecules and create a new topology file
strip :WAT nobox parmout nowater_dimer_c_amber_solv_1264_wt_I104F.prmtop

# Fit the structures by removing global movements (translation and rotation)
rms fit :*

# Export the processed trajectory without water
trajout nowater_wt_I104F_md_trajectories.nc

run
clear all

# Load the topology file without water and with the I104F monomer
parm nowater_dimer_c_amber_solv_1264_wt_I104F.prmtop

trajin nowater_wt_I104F_md_trajectories.nc

strip :Na+,Cl-
cluster cl \
  kmeans clusters 10 randompoint maxit 500 \
  rms @C,N,O,CA,CB&!@H= \
  sieve 10 random \
  out cnumvtime.dat \
  summary summary.dat \
  info info.dat \
  cpopvtime cpopvtime.agr normframe \
  repout rep repfmt pdb \
  singlerepout singlerep.nc singlerepfmt netcdf \
  avgout avg avgfmt pdb
run
```

AUTODOCK VINA configuration file

```
center_x = 0.0
center_y = 0.0
center_z = 0.0

size_x = 80
size_y = 80
size_z = 80

num_modes = 10
energy_range = 4
exhaustiveness = 32
```

Batch script 1: running VINA multiple times

```
@echo off
setlocal EnableDelayedExpansion

:: Richiedi all'utente il numero di iterazioni
set /p iterations=Inserisci il numero di iterazioni:

:: Configura le variabili comuni
set "receptor=recettore.pdbqt"
set "ligands_dir=ligandi"
set "config_file=config.txt"
set "summary_dir=summary_logs"

:: Controlla se il recettore esiste
if not exist "%receptor%" (
    echo ERRORE: Il file del recettore "%receptor%" non esiste.
    pause
    exit /b
)

:: Controlla se la directory dei ligandi esiste
if not exist "%ligands_dir%" (
    echo ERRORE: La directory dei ligandi "%ligands_dir%" non esiste.
    pause
    exit /b
)

:: Crea la directory per i file di riepilogo se non esiste
if not exist "%summary_dir%" mkdir "%summary_dir%"

:: Loop attraverso il numero di iterazioni specificato
for /l %i in (1,1,%iterations%) do (
    echo Inizio iterazione %i...

    :: Configura le variabili per l'iterazione corrente
    set "iter_dir=iterazione_%i"
    set "output_dir=!iter_dir!\output"
    set "log_dir=!iter_dir!\logs"
    set "filtered_log_dir=!iter_dir!\filtered_logs"
    set "summary_file=!iter_dir!\top_affinity_data.txt"

    :: Crea la directory dell'iterazione e le sottodirectory se non esistono
    if not exist "!iter_dir!" mkdir "!iter_dir!"
    if not exist "!output_dir!" mkdir "!output_dir!"
    if not exist "!log_dir!" mkdir "!log_dir!"
    if not exist "!filtered_log_dir!" mkdir "!filtered_log_dir!"

    :: Elimina il file di riepilogo se esiste già
    if exist "!summary_file!" del "!summary_file!"

    :: Loop attraverso ogni ligando
    for %f in ("%ligands_dir%\*.pdbqt") do (
        set "ligand=%f"
        set "ligand_name=%~nf"
        set "output_file=!output_dir!\docked_%~nf.pdbqt"
        set "log_file=!log_dir!\log_%~nf.log"

        echo Inizio docking per ligando: %f
        echo Comando: vina --receptor "%receptor%" --ligand "%f" --out "!output_file!" -
-log "!log_file!" --config "%config_file%"

        :: Esegui AutoDock Vina
        vina --receptor "%receptor%" --ligand "%f" --out "!output_file!" --log
"!log_file!" --config "%config_file%"

        :: Controlla se il file di output è stato creato
```

```

if not exist "!output_file!" (
    echo ERRORE: Docking fallito per %%f. Il file di output non è stato creato.
    pause
    exit /b
) else (
    echo Docking completato per %%f
)

:: Controlla se il file di log è stato creato
if exist "!log_file!" (
    set "count=0"
    set "lineCount=0"

    :: Conta il numero totale di righe nel file di log
    for /f "tokens=*" %%1 in (!log_file!) do (
        set /a "lineCount+=1"
    )

    :: Legge il file di log e salva solo le righe dalla 25ª alla penultima
    > "!filtered_log_dir!\log_%%~nf_filtered.txt" (
        set "count=0"
        for /f "tokens=*" %%1 in (!log_file!) do (
            set /a "count+=1"
            if !count! geq 25 (
                if !count! lss !lineCount! (
                    echo %%1
                )
            )
        )
    )

    echo          File          di          log          filtrato          salvato          come
"!filtered_log_dir!\log_%%~nf_filtered.txt"

    :: Salva la 25ª riga nel file di riepilogo
    set "count=0"
    for /f "tokens=*" %%1 in (!log_file!) do (
        set /a "count+=1"
        if !count!==25 (
            echo %%1 >> "%summary_dir%\docking_%%~nf.txt"
            echo %%1 >> "!summary_file!"
        )
    )
) else (
    echo ERRORE: Il file di log "!log_file!" non esiste.
)

echo Iterazione %%i completata.
)

echo Docking completato per tutte le iterazioni.
Pause

```


Python script 2: RMSD matrix of docking poses

```
import numpy as np
import MDAnalysis as mda
from MDAnalysis.analysis import rms
import os
import re
import warnings

# Suppress warnings related to atom types
warnings.filterwarnings("ignore", category=UserWarning)

# Function to find PDBQT files in the current folder
def find_pdbqt_files():
    # Gets the folder where this script is located
    current_directory = os.path.dirname(os.path.abspath(__file__))
    # Finds all .pdbqt files in the folder
    pdbqt_files = [f for f in os.listdir(current_directory) if
f.endswith('.pdbqt')]

    # Function to extract the number after "iter_" in the filename
    def extract_iter_number(filename):
        match = re.search(r'iter_(\d+)', filename)
        return int(match.group(1)) if match else float('inf') # If it doesn't
find "iter_", returns a very high number

    # Sorts files based on the extracted number
    sorted_files = sorted(pdbqt_files, key=extract_iter_number)

    return [os.path.join(current_directory, f) for f in sorted_files]

# Function to extract models from a PDBQT file
def extract_models(file):
    models = []
    model_lines = []

    with open(file, 'r') as f:
        for line in f:
            if line.startswith("MODEL"):
                # New model found, if there are lines from the previous model,
save them
                if model_lines:
                    models.append(''.join(model_lines))
                    model_lines = [] # Clears for the new model
                model_lines.append(line)

            if line.startswith("ENDMDL"):
                # End of the model, save the current model
                models.append(''.join(model_lines))
                model_lines = [] # Clears for the next model

    return models

# Function to calculate RMSD between two models
def calculate_rmsd(model1_str, model2_str):
    # Creates temporary files for the models
    with open("temp_model1.pdbqt", "w") as f:
        f.write(model1_str)
    with open("temp_model2.pdbqt", "w") as f:
        f.write(model2_str)
```

```

# Creates MDAnalysis universes from the temporary files
u1 = mda.Universe("temp_model1.pdbqt", format='PDBQT')
u2 = mda.Universe("temp_model2.pdbqt", format='PDBQT')

# Calculates the RMSD
rmsd_value = rms.rmsd(u1.atoms.positions, u2.atoms.positions)

return rmsd_value

# Function to create the RMSD matrix
def create_rmsd_matrix(all_models, model_labels):
    total_models = len(all_models)
    rmsd_matrix = np.zeros((total_models, total_models))

    for i in range(total_models):
        for j in range(i, total_models):
            if i == j:
                rmsd_matrix[i][j] = 0.0 # The main diagonal is 0
            else:
                rmsd_matrix[i][j] = calculate_rmsd(all_models[i], all_models[j])
                rmsd_matrix[j][i] = rmsd_matrix[i][j] # Symmetry

    return rmsd_matrix

# Main function
def main():
    print("Searching for PDBQT files in the current folder...")
    pdbqt_files = find_pdbqt_files()

    if len(pdbqt_files) == 0:
        print("No PDBQT files found in the current folder.")
        return

    all_models = []
    model_labels = [] # List for model names

    print(f"Loading models from {len(pdbqt_files)} PDBQT files...")

    # Extract models from each PDBQT file
    for file in pdbqt_files:
        print(f"Extracting models from file: {file}...")
        models = extract_models(file)
        all_models.extend(models) # Add models to the complete list
        model_count = len(models)
        model_labels.extend([os.path.basename(file) + f" (Model {i+1})" for i in
range(model_count)])

    print(f"Total models loaded: {len(all_models)}")

    if len(all_models) < 2:
        print("Select at least two models to calculate the RMSD matrix.")
        return

    # Calculate the RMSD matrix
    rmsd_matrix = create_rmsd_matrix(all_models, model_labels)

    # Normalize the matrix by setting the diagonal to 1
    np.fill_diagonal(rmsd_matrix, 1)

    # Print the RMSD matrix
    print("RMSD Matrix:")
    print(rmsd_matrix)

```

```

# Save the matrix in a file
output_file = "rmsd_matrix.txt"
np.savetxt(output_file, rmsd_matrix, fmt="%.3f")
print(f"RMSD Matrix saved in {output_file}")

# Print the model row/column correspondences
print("\nRow/Column to Model Correspondence:")
for idx, label in enumerate(model_labels):
    print(f"Row/Column {idx}: {label}")

# Remove temporary files
os.remove("temp_model1.pdbqt")
os.remove("temp_model2.pdbqt")
print("Temporary files removed.")

# Pause to keep the window open
input("\nPress Enter to exit.")

if __name__ == "__main__":
    main()

```

MATLAB script 3

```
clc;
clear;
close all;

% Number of ligands
num_ligands = 35;

% Preallocate cells to save variables for each ligand
binsizes = cell(1, num_ligands);
bins = cell(1, num_ligands);
vina_scores_v = cell(1, num_ligands);
mean_vina_scores = cell(1, num_ligands); % To store the mean scores for each
cluster

% Loop over each ligand
for i = 1:num_ligands

    % Construct the folder name for each ligand (with underscore)
    folder_name = ['ligand_', num2str(i)];

    % Load the RMSD matrix from the current folder
    rmsd_file = fullfile(folder_name, 'rmsd_matrix.txt');
    rmsd_matrix = load(rmsd_file);

    % Load the VINA scores from the current folder
    vina_scores_file = fullfile(folder_name, 'vina_scores.txt' );
    vina_scores_v{i} = load(vina_scores_file);

    %% Algorithm: Connected components
    threshold_connected = 4; % Angstrom threshold

    % Create a binary adjacency matrix (0 if RMSD >= threshold, 1 if RMSD <
threshold)
    adj_matrix = rmsd_matrix < threshold_connected;

    % Set the diagonal to zero (no self-similarity)
    for j = 1:size(adj_matrix, 1)
        adj_matrix(j, j) = 0;
    end

    % Create a graph from the binary adjacency matrix
    G = graph(adj_matrix);

    % Find the connected components (the clusters)
    [bin, binsize] = conncomp(G);

    % Save the pose-cluster associations
    bins{i} = bin;

    % Save the cluster sizes
    binsizes{i} = binsize;

    % Calculate the mean Vina score for each cluster
    num_clusters = max(bin); % Number of clusters
    mean_vina_scores{i} = zeros(1, num_clusters); % Preallocate the mean score
vector
    for cluster_id = 1:num_clusters
        % Find the indices of the poses belonging to the current cluster
        poses_in_cluster = find(bin == cluster_id);
        % Calculate the average Vina scores for these poses
```

```

        mean_vina_scores{i}(cluster_id)
mean(vina_scores_v{i}(poses_in_cluster));
    end
end

% Prepare the data for the plot
all_sizes = []; % Array for cluster sizes
ligand_labels = []; % Array for ligand labels
cluster_labels = []; % Array for cluster labels
all_mean_vina = []; % Array for the mean Vina scores

% Loop to gather data for the plot
for i = 1:num_ligands
    if ~isempty(binsizes{i})
        for j = 1:length(binsizes{i})
            all_sizes(end + 1) = binsizes{i}(j); % Add the cluster size
            ligand_labels(end + 1) = i; % Add the ligand
            cluster_labels(end + 1) = j; % Add the cluster number
            all_mean_vina(end + 1) = mean_vina_scores{i}(j); % Add the mean Vina
score
        end
    end
end

% Create the scatter plot
figure;
hold on;

% Create a color map (e.g., from blue to red)
colormap jet;
c = colorbar; % Adds a color bar
c.Label.String = 'Mean Vina Score';

% Set the color of the description and color bar labels to white
c.Label.Color = 'w'; % Color bar label
c.TicksMode = 'auto'; % Keep the automatic ticks
c.Color = 'w'; % Color of the tick values in the color bar

% Create the scatter plot with colors depending on the mean Vina score
for i = 1:length(all_sizes)
    scatter(ligand_labels(i), all_sizes(i), 30, all_mean_vina(i), 'filled'); %
Color based on Vina score
end

xlabel('Ligands', 'Color', 'w'); % White text
ylabel('Cluster Sizes (Number of Poses)', 'Color', 'w'); % White text
title('Cluster Sizes and Mean Vina Scores for Each Ligand', 'Color', 'w'); % White
text

% Set the x-axis labels
xticks(1:num_ligands);
xticklabels(arrayfun(@(x) num2str(x), 1:num_ligands, 'UniformOutput', false)); %
Only ligand numbers
grid on;

% Set the background to black
set(gca, 'Color', 'k');
set(gcf, 'Color', 'k'); % Set the figure color to black

% Set the axis label colors and tick colors
set(gca, 'XColor', 'w', 'YColor', 'w'); % Axis labels in white
hold off;

```