# Computational analysis of the three dimensional structure of the human Haemoglobin protein

**DANIELE TRAVERSA**[1]

[1] *Alma Mater Studiorum, University of Bologna*
[2] *Department of Pharmacy, Biotechnology and Sport Science*
[3] *First Cycle degree in Genomics(A.Y. 2019-2020)*
[4] *Corresponding author: daniele.traversa@studio.unibo.it*

*Compiled February 13, 2020*

**Haemoglobin, a tetrameric protein made of 4 monomers, two subunits alpha and two subunits beta whose role is to transport oxygen in the blood circulation. In the following paper we explain how we have employed different computational strategies to analyse its molecular structure. We started selecting the protein from the Protein Data Bank. Subsequentially, we developed a program that was able to retrieve peculiar information regarding the protein. In our case we were interested in identifying the atoms that granted the interactions between each monomer of the protein and the interaction between the monomers and the haem and oxygen atoms that has been crystalized and consequentially inserted in the pdb file. Then, we wanted to highlight the interaction surface between each monomer. To do so, we used a program to convert a pdb file into a DSSP file and per each residue of each monomer, we identified the change of relative solvent accessible area when the monomer passes from being alone in space to being part of the haemoglobin complex.**
**To conclude, we decided to implement a program that was able to identify the subnetwork of proteins which were somehow in contact with the subunit alpha and the subunit beta of the haemoglobin protein. The subnetwork included those proteins having a maximum path length of two with respect to the two proteins, so the proteins that either directly interacted with subunit alpha or subunit beta or interacted by means of another protein so indirectly.**

http://dx.doi.org/10.1364/ao.XX.XXXXXX

## 1. INTRODUCTION

Haemoglobin is an oxygen-transport protein containing four haem prosthetic groups. It is mainly operative in the red blood cells of almost all vertebrates as well as in the tissues of some invertebrates. Haemoglobin has the role of carrying oxygen from the lungs to the rest of the cells of the other tissues. Here, it releases oxygen that allows the other cells to perform aerobic respiration that is fundamental process to produce energy. The latter is used by the cell in the different metabolic pathways, grating the sustain of the organism. Once oxygen has entered the blood from the lungs, it is taken up by haemoglobin in the red blood cells [3]. When the protein molecule chemically reacts with oxygen, it forms oxyhaemoglobin. Haemoglobin changes shape based on how many oxygen molecules are bound to it. The change in shape also causes a change in affinity to oxygen; the more oxygen is bound, the higher the molecule's affinity for oxygen becomes. This is known as cooperative binding. When no oxygen is bound, the haemoglobin is said to be in the Tense State (T-state), with a low affinity for oxygen. At the point where oxygen first binds, the haemoglobin alters its shape into the Relaxed State (R-state), which has a higher affinity for oxygen. The oxygen delivery in the environment is related to the oxygen partial pressure pO2 that is present in the tissue. Hence, we observe that when the oxygen partial pressure is high, the oxygen saturation level of haemoglobin is high. This situation is observed when haemoglobin is in blood circulation system where the concentration of oxygen is higher and the protein immediately reaches the maximum saturation level. Otherwise, when the oxygen partial pressure is low, we observe that haemoglobin has a low oxygen saturation level. This situation is observed in many tissues where haemoglobin arrives through the blood circulation and due to the low oxygen pressure, releases oxygen in the environment. The mammalian haemoglobin molecule is able to carry up to four oxygen molecules that bind to the prosthetic groups of the protein, the haem groups through an iron atom Fe2+ [7] [8].

Haemoglobin is a tetrameric protein composed of two different types of monomers: two alpha subunits that we

denoted as chain A and chain C, and two beta subunits that we named as chain B and chain D. Each of the is provided of a haem group. For this reason, one haemoglobin protein is able to transport 4 oxygen atoms.

### A. Subunit alpha

The tetramer is composed of two different alpha subunits that are almost identical to each other: chain A and chain C. Both chains have a length of around 141 amino acids. In humans, the two monomers are encoded by two different genes called HBA1, for chain A, and HBA2 for chain C that are paralogues to each other. They are localized in the alpha-globin gene cluster that is located at the very tip of the short arm of chromosome 16. The two coding sequences are identical, suggesting that they are highly conserved due to their fundamental function. These genes differ slightly over the 5′ untranslated regions and the introns, but they differ significantly over the 3′ untranslated regions [1] [2] [3].

**Figure 1: 3D structure of subunit alpha of haemoblobin**

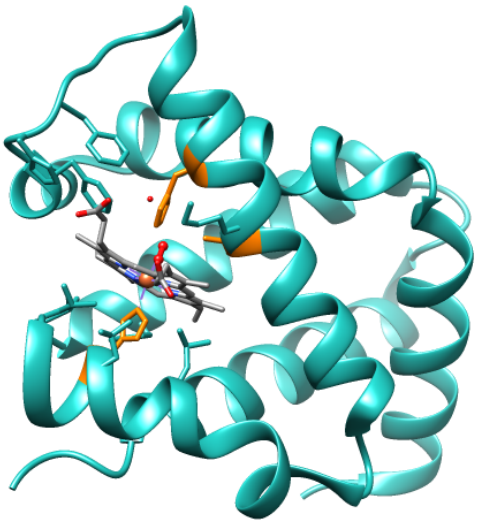


*Figure 1 reports the subunit alpha (chain A) of haemoglobin tetramer. It has been obtained using the visualization software UCSF CHIMERA.*

### B. Subunit Beta

In the case of the beta subunit, we can distinguish two different monomers that are identical: chain B and chain D. The two chains count of 146 amino acids and are encoded by a unique gene called HBB that is found within a 45 kb cluster on chromosome 11. Two subunit alpha plus two subunit beta form a tetramer called HBA1 that represent the 97% of the total haemoglobin present in adult humans. The remaining 3% is constituted by another tetramer, called HBA2, that is made of two alpha subunits and two delta subunits. The latter are encoded by another gene called HBD that is found in the same cluster where the HBB gene is found [1] [2] [4].

The genes for the protein chains of hemoglobin show small differences within different human populations, so the amino acid sequence of hemoglobin is slightly different from person to person. In most cases the changes do not affect protein function and are often not even noticed. However, in some cases these different amino acids lead to major structural changes. One such example is that of the sickle cell hemoglobin, where a glutamate in the beta chain is mutated to valine. This deforms the red blood cell, which is normally a smooth disk shape, into a C or sickle shape [5].

**Figure 2: 3D structure of subunit beta of haemoblobin**

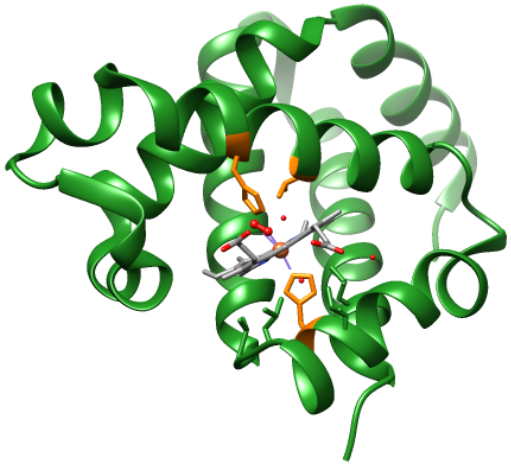


*Figure 2 reports the subunit beta (chain B) of haemoglobin tetramer. It has been obtained using the visualization software UCSF CHIMERA.*

### C. Three Dimensional Structure

Looking at the secondary structure image provided by the PDB, Protein Data Bank, we observe that most of the amino acids in haemoglobin form alpha helices, and these helices are connected by short non-helical segments. Hydrogen bonds stabilize the helical sections inside this protein, causing attractions within the molecule, which then causes each polypeptide chain to fold into a specific shape. Haemoglobin's quaternary structure comes from its four subunits in roughly a tetrahedral arrangement. This kind of fold is called, globin fold arrangement. A haem group consists of an iron (Fe) ion held in a heterocyclic ring, known as a porphyrin. This porphyrin ring consists of four pyrrole molecules cyclically linked together (by methine bridges) with the iron ion bound in the centre. The iron ion lies of the plane provided by these groups and it is responsible for oxygen binding [5].

**Figure 3: 3D structure of haemoglobin**

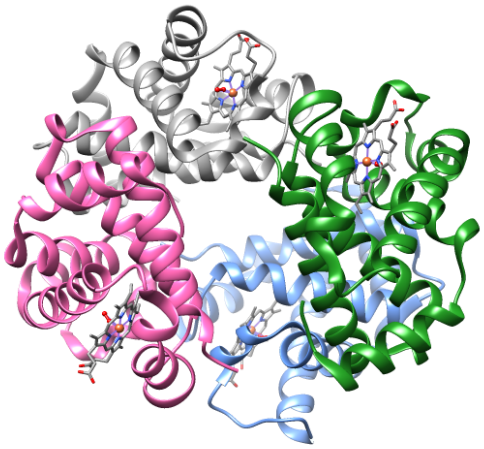


*Figure 3 reports the haemoglobin tetramer. It has been obtained using the visualization software UCSF CHIMERA. Each chain is reported with a different colour: chain A(blue);chain B(green);chain C(grey);chain D(pink).*

## 2. METHODS AND MATERIALS

The study has been conducted using different computational tools that allowed us to perform the analysis in a more accurate way. To make the analysis clearer, we divided it in three sections. Per each section we describe the tools that we used and the pipeline that we conducted.

### A. Physical Interactions

In this part of the analysis we used different software's, databases and computational tools. In first place, we worked with the Protein Data Bank(PDB). From such, we retrieved the PDB file for the haemoglobin protein. The PDB identifier is 1GXZ.

To develop the programs needed for our analysis, we adopted the Python programming language(Version 2.7) and additional Python packages.

Finally, to visualize the 3D structure of haemoglobin, we employed UCSF Chimera. Basically, is a highly extensible program for interactive visualization and analysis of molecular structures and related data, including density maps, supramolecular assemblies, sequence alignments, docking results, trajectories, and conformational ensembles.

All these instruments has been exploited on a UNIX operating system which has been chosen for its dynamism and speed in computing the results.

The experiment began when we took in consideration of the haemoglobin structure from the Protein Data Bank. The latter is one of the major protein databases containing the 3D structure of proteins that has been retrieved through X-ray crystallography, NMR or other techniques. Each protein is associated to a PDB identifier that univocally distinguish one protein and allow us to directly move to the section of the database that is dedicated to the protein. Here we find many subsections that describe the protein in detailed way like the content about the protein in literature, the macromolecules composing the protein the micro molecules composing the protein and the 3D view of the protein.

Otherwise all structural information about how the protein is organized in the 3D space are reported in PDB file (protein.pdb) that basically contains all the molecular evidences that has been retrieved in the 3D structure obtained through X-ray crystallography, NMR or other techniques.

The PDB file is organized in a column-like structure where each row represents an atom while each column highlights a specific feature per each atom like the amino acid number to which the atom belong to, the kind of atom we are dealing with, the amino acid, the chain where this atom is found, the X,Y, and Z coordinates of the atom in space and the resolution of the 3D structure obtain in the experiment.

The PDB file can be download directly from the Protein Data Bank and can be used for secondary analysis.
In our experiment we considered a specific structure of haemoglobin which is the oxy T state haemoglobin that basically represents the tetramer where the four haem groups are bound to oxygen (PDB identifier: 1GXZ [6]). This structure has been obtained through X-ray crystallography and the level of resolution that has been reached is 2.1A.
Considering that in our structure we also have the presence of other atoms, i.e the haem groups and the oxygen, that are external to the amino acid sequence, in the pdb file they are reported as HETATM.

In the first part of our experiment, we downloaded the pdb file. We did not needed any additional permission since the PDB database is freely accessible and allows the user to download the pdb file without any constriction.
After that we used the python programming language to create a program that was able to parse the pdb file and to retrieve the information that we were interested in.
In our case, we wanted to see how the four monomers of the protein interacted with each other and how they interacted with the haem groups and the oxygen. So, at the beginning, we concentrate ourselves on the physical interactions between each monomer and then between each monomer and the oxygen groups and the haem groups.
To see if two residues interacted, we calculated the distance between each pair of atoms. In the moment in which we find two atoms that are located at a distance of 3.5A°, the two residues interact. We set this threshold because, looking at different sources, we understood that the optimal distance between two atoms to create a hydrogen bond is when they are at most 3.5A° apart [9]. In our case, we are dealing with putative hydrogen atoms since the PDB file does not contain them.

The python script that we developed was made of different functions each covering a specific role in the analysis of the pdb file.

A function was aimed to parse the pdb in a way to collect only the information regarding the atoms and their position in the three-dimensional space so the X, Y and Z coordinates. In the function, we opened the pdb file and we parsed it.

Meanwhile, using the same script, we created a data structure, using different python objects, that was able to accommodate the coordinate and the name of each atom per amino acid per chain. This function was also used to retrieve this information also for the haem group and the oxygens per each chain.

Then, using another function in the python program, we calculated the distance between each pair of atoms between two residues. In the moment in which two atoms have a distance lower then 3.5A, we assumed that the two atoms interacted. The calculation of the distance was done using a python package called Numpy [16].

We used the visualization program called CHIMERA in a way to filter out those putative interactions that are more likely to clearly form hydrogen bonds in the tetramer [12]. Basically, once we retrieved the atoms, we loaded the pdb file into chimera. Then we selected each pair of monomers and per each we looked at the orientation of the atoms between the side chains of the amino acids that, according to our results, should interact.

The script is provided of supplementary functions that cluster the interactions depending on different criteria to distinguish the interactions between haem groups and oxygen groups with the monomers of the haemoglobin protein and the interactions between each monomer.

The same procedure has been conducted considering the interaction between the residues of each monomer and the haem groups and the oxygen that are included in the PDB file. The goal was to identify which residues stabilize the interaction between the monomer and the haem prosthetic groups and the oxygen groups.

### B. Surface Interactions

In this section, we used different software's, databases and computational tools.
Firsly we have to mention the DSSP program. It is a software able to calculate and assign the most likely secondary structure of a protein given the 3D structure. It does this by reading the position of the atoms in a protein followed by calculation of the H-bond energy between all atoms. The algorithm will discard any hydrogens present in the input structure and calculates the optimal hydrogen positions by placing them at 1.000 Å from the backbone N in the opposite direction from the backbone C=O bond.[11].

We used this program to create different DSSP files. The original PDB file has been divided in sub PDB each contains monomers (one per each monomer) and trimers (one per each combination of three monomer). Per each PDB we used the DSSP program to get different DSSP files that we used to understand the surface interactions within the tetramer.

The secondary structure has been analyzed employing a script written in Python, a programming language(version 2.7). We needed additional python libraries to address the problem we were dealing with.

All these instruments has been exploited on a UNIX operating system too.

In the second part of our experimental analysis, we focused on identifying the interaction surface between the monomers of the protein in the moment in which they form a tetramer. Moreover, we also analysed the loss of solvent accessibility per each residue in a way to find which residues were clearly interacting through hydrophobic interactions.
The goal was to identify how the four monomers interacted, so we wanted to quantify the surface of interaction between each pair of monomer and the interaction hot-spots which represent those amino acids residues that loose a high value of relative solvent accessibility when they pass from being alone in space to being part of the tetramer. Basically, if a residue has a high solvent accessibility, it means that when it is located inside a solvent, the majority of the surface of this amino acid is in contact with the solvent. This situation is observed in polar residues. Otherwise, when a residue has low solvent accessibility it means that, when it is exposed to the solvent environment, it poorly faces it. This situation is preferably observed when two hydrophobic residues interact. Considering that when we have a multi-monomeric protein the monomers usually interact thought hydrophobic interaction, our goal is to find those residues that significantly loose solvent accessibility in the tetramer because they will represent the amino acids that grant the interaction between two monomers in the complex.

In first place, we needed a way to retrieve the major information about the secondary structures of the four chains present in the tetramer. To do so, we used a DSSP program that is a software able to take the pdb file as input and to calculate the different features of a secondary structure and put them in a DSSP file. The information that this program is able to calculate are of different kind such as the secondary structure where the amino acid is found, the solvent accessibility of the amino acid,the Phi angle and the Psi angle.

We imported the DSSP program from an online Github repository and after we installed it on our machines, we created different kind of DSSP files. The DSSP file is organized in a column like structure where each row is an amino acid and each column explains different features of the residue when it is fold in a secondary structure like the residue number, the chain where it belongs to, the amino acid type, the secondary structure; H(helix),T(turn),E(extended), the Phi and Psi angles and the solvent accessible area (A°2)

Firstly, we consider the entire tetramer: using the command line on the Linux terminal we grouped all the atoms in a new pdb file and we run the DSSP program. The results were then collected in a DSSP file that contains all the information of the secondary structures of the entire tetramer.
The same process have been performed considering the four chains alone: therefore using again the command of the UNIX terminal, we were able to separate each of the four chains in four new pdb files and to create four different DSSP files, each representing the secondary structure of each monomer alone in space.
To conclude we did the same pipeline considering all possible combination of trimers that can be extracted from the tetramer.
Why do we needed the trimer too?
Basically, in the moment in which we wanted to calculate the interaction surface between two monomers, we needed to evaluate the solvent accessibility of each residue of a peculiar monomer when the latter was in the complex and when it

was in a trimer where the other monomer, with which it was supposed to interact, was missing.
In this way we were able to see if there was a significant loss of solvent accessibility when the two monomers were interacting and when not. Hence, if the loss is high, means that the two monomers strongly interact.
Therefore, at the end of this procedure, we have understood which monomers interact with each other and how strong is the interaction was.
However, the absolute solvent accessible area does not tell us which portion of the amino acids is actually exposed to the solvent and which not.
Hence, we calculated the relative solvent accessible area per each amino acid. When we observed a high loss in the relative solvent accessibility of the residue in a chain when it was alone in space and when it was in the complex, the amino acid was involved in the interaction between two monomers.

These pipelines have been conducted using the Linux terminal commands and we developed a python script that was made of different kind of functions, each playing a specific role.

The major function was the one used to parse the DSSP file. Its goal was to extract the solvent accessibility area of each amino acid in the DSSP file. The function has been used for each DSSP file we created: monomer, trimer and tetramer.

Then, another function was entitled to calculate the absolute accessible area of a monomer when it was part of the tetramer and when it was part of trimer where another monomer was missing. Calculating the difference in accessibility area when the monomer was in the trimer and when it was in the complex, we quantify the surface interaction between each pair of monomers.

Then, to calculate the relative solvent accessible area of each residue, we used a supplementary data-frame containing the estimate of the maximum accessible surface of each amino acid. We divided the accessible surface of the residue in the DSSP file over the its maximum accessible surface. We calculate the relative accessible area for all the amino acids of each chain when the latter was alone in space and when it was part of the complex. After that, we calculated the difference between the relative solvent accessibility of an amino acid. If we registered a change, so a loss of solvent accessibility, that was higher the 10%, the residue was involved in the interaction between two monomers.

The script is also provided of supplementary functions that better organize the information and the results that we obtain performing the different computational experiments.

In this python program we used the sys library as in the previous program we have explained in the article.

### C. Protein-Protein Interaction Network

In the third step, we focused in the analysis of the protein interaction network of homo sapiens. In particular, we wanted to study the interactome of the two subunits of haemoglobin: subunit alpha and subunit beta.
To achieve this, we used different softwares, databases and computational tools. First of all we employed the Intact EMBL-EBI database. It is a database that provides a freely available, open source database system and analysis tools for molecular interaction data. All interactions are derived from literature curation or direct user submissions and are freely available [13].

To develop the programs for the analysis we adopeted the Python programming language(version2.7) and additional Python packages. The most important was networkX, a library that provides several tools for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks. [15]

Then, we performed a statistical analysis of the results that we obtained using the R programming language (version 3.5.1). We decided to exploit the R programming language since it is optimized for statistical computing and for data visualization.

In the first step of our study, we downloaded a file from the Intact database containing all possible pairwise interaction between all known proteins. Per each pairwise interaction, the file reports many information like the Uniprot id of the two proteins that interact, the experiment through which this interaction has been confirmed (i.e X-ray crystallography. NMR ect.)and the taxonomy id of the organism in which this interaction has been found.

Considering that we are interested in the interaction involving the subunit alpha and the subunit beta of human haemoglobin, we needed to parse the Intact file in a clever way. As a consequence, we developed a python program to do so.
This python script is made of different function, each responsible to provide us a specific information.

To start, we developed a function that took the Intact interactome file as input and it collected the binary protein interaction in a python data frame. Each element of the data frame contained specific notions like the first protein involved in the interaction, the second protein involved in the interaction and the respective taxonomy id for the two proteins. These elements are fundamental to characterize each interaction.

Each protein was identified by means of the uniprot id.
Then, since we are focused in the human interactome, we used two other function though which we filtered out the pairwise interactions that we retrieved in the primary analysis of the Intact Database file.

One function had the role to retrieve only the protein interactions that were found in the homo sapiens. So looking at the taxonomy id, we considered only those pairwise interactions of the human proteome.

Then, since these interactions may be redundant, we used another function to get only the unique one. Basically, we took the human-filtered data frame containing the pair wise interactions and we created a new python data frame where we inserted the pairwise interaction identified and the number of times this interaction is reported in the Intact database file.

So, after all these procedures, we ended with the entire human proteome. However, we were focused on the protein

interacting with the subunit alpha and the subunit beta. Hence we needed to create a sub graph were we considered only the protein that interact with subunit alpha or subunit beta with a maximum path of two which means that we retrieved the protein that interact directly with subunit alpha or beta or at least interact with them by mean of another protein. We decided to analyse the interactome in this way because, otherwise, we could not analysis the protein interactions due to the high number of computations.

To perform this procedure, we used another function that was also able to perform the statistical analysis of the results that we got. Basically, per each protein that directly interacts with the subunit beta or subunit alpha that we extracted, we calculate three fundamental quantities of graph theory.
Degree. Quantity associated to each node. It indicates the number of edges adjacent to the node, so the number of edges connected to a node.
Clustering coefficient; It defines the number of edges attached to the current node over the number of edges connected to all the neighbour's nodes with respect to the current one. Basically, it defines locally the number of connections referred to a node with respect ft the number of connections associated to the nodes surrounding the current one. These elements quantifies, locally, the importance of a node.
Betweenness centrality. It indicates the number of shortest paths between two nodes, I and j, passing through a current node k over the number of all possible shortest paths between I and j. As a consequence, it quantifies the importance of a node in the network.

So at the end of the analysis, we obtained a table containing the proteins that have a direct interaction with one of the two subunits and per each of them we reported the degree, clustering coefficient and betweenness centrality.

## 3. RESULTS

The results that we obtained from our experiments has been collected in tables. We created a total number of four tables, one per each analysis conducted.

### A. Monomer-Monomer Physical Interactions

| Chain1 | Res 1 | Chain2 | Res 2 | Atoms($\leq$3.5Å) |
|---|---|---|---|---|
| A | ASP526 | C | ARG141 | OD2-CZ,OD2-NH1,OD2-NH2 |
| A | LYS527 | C | ARG141 | NZ-C,NZ-OXT,NZ-O,CE-O,CD-O |
| A | ARG541 | C | LYS127 | C-NZ,OXT-NZ,O-NZ |
| A | ARG541 | C | ASP126 | CZ-OD2,NH1-CG,NH1-OD2,NH2-OD2 |

*Table 1: Putative interacting Residues between chain A and chain C*

In this table, we reported the residues that should interact between each monomer through hydrogen bonds. Basically, the pdb file does not contain the hydrogen hence the atoms that we found to interact, are interacting through putative hydrogen bonds. So, to assess the real presence of them between two residues, we used the visualization software called Chimera. We selected the two atoms that have a distance between 3.5A° from each other and we checked if the orientation of the residues was such to grant the formation of the bond. With this procedure, we found some atoms that, even if are localized 3.5A°, were oriented in a way to not favouring the formation of such bonds, hence they have a lower probability to interact.
Additional tables are reported in the supplementary materials.

Using this process, we identified also salt bridges that, by definition, are interactions between two charged residues that posses opposite charge. Salt bridges are important to stabilize the interactions between two monomers[10]. The salt bridges that were found are between ASP526-ARG141, ARG541-ASP126 and GLU86-ARG92

**Figure 4: Amino Acids forming putative salt bridges**

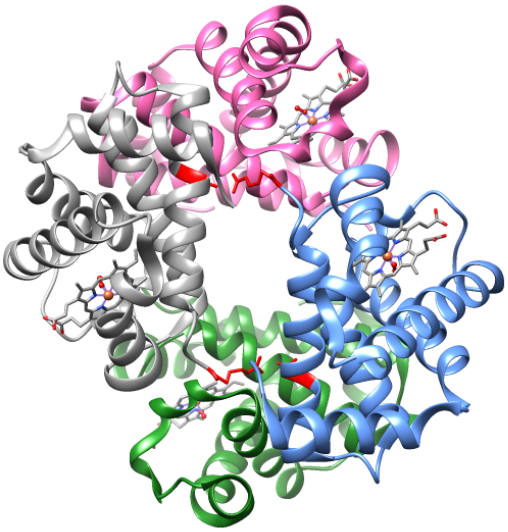


*Figure 4 reports the putative salt bridges that could be formed to stabilize the tetramer. They are colored with red. It has been obtained using the visualization software UCSF CHIMERA.*

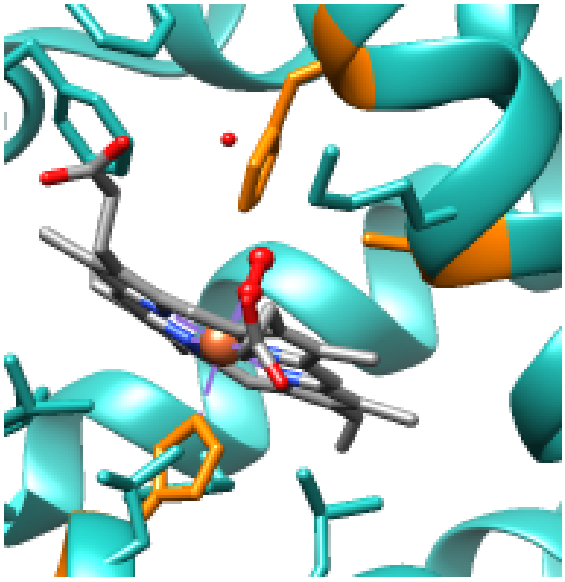
### B. Monomer-Haem-Oxygen Interactions

| Chain | Residue | Heteroatom | Atoms(≤3.5Å) |
|---|---|---|---|
| A | TYR42 | HEM1142 | O-CMD |
| A | PHE43 | HEM1142 | CE2-CMD,CZ-CHD |
| A | ASN97 | HEM1142 | O-CMC,CE-O1A |
| A | LYS61 | HEM1142 | O-CMA |
| A | PHE46 | HEM1142 | CE2-O1D |
| A | HIS87 | HEM1142 | CE1-FE,CD2-FE,NE2-FE,CE1-NA,NE2-NA,CE1-NB,NE2-NB,CD2-NC,NE2-NC,NE2-ND,CE1-C4A |
| A | VAL93 | HEM1142 | CG1-CAC |
| A | HIS58 | HEM1142 | CE1-C4D,CE1-C1A |
| A | LEU91 | HEM1142 | CD1-C4D |
| A | PHE98 | HEM1142 | CE1-CHC,CD1-CBB |
| A | HIS45 | HEM1142 | NE2-O2D |
| A | LEU86 | HEM1142 | CD2-CBA |
| A | LEU83 | HEM1142 | CD1-C3A |
| A | HIS58 | OXY1143 | NE2-O2,NE2-O1 |
| A | VAL62 | OXY1143 | CG2-O2 |

*Table 2: residue-heteroatom interactions of chain A*

The following tables represent the interaction that we found between each monomer and the haem groups and the oxygen groups.
Basically, in these tables we reported the amino acids that, in each monomer, create hydrogen bonds with the haem group and the oxygen.
In our python programs, we set that, in the moment in which the distance between two pairs of atoms between two residues was below 3.5A°, the two atoms (a donor and an acceptor) interact through hydrogen bonds.
We had to set this constriction because hydrogen atoms are not reported in the pdb file and using the proper documentation, we find that when the distance between the donor and the acceptor is below 3.5A°, it is probable that the two atoms interact forming a hydrogen bond. To see the results for the other chains, consult the supplementary materials.

**Figure 5: Chain A and haem group interraction**



*This picture shows the interaction between the chain A, so a subunit alpha and the haem group and the oxygen atom*

### C. Monomer Surface Interaction

| Chain1 | Chain2 | SA(Å)1-2 | SA(Å)2-1 |
|---|---|---|---|
| A | B | 850.0 | 886.0 |
| A | C | 241.0 | 238.0 |
| A | D | 697.0 | 671.0 |
| B | C | 659.0 | 670.0 |
| B | D | 24.0 | 22.0 |
| C | D | 809.0 | 826.0 |

*Table 3: Monomer surface interaction table.*

The following table illustrates the results that we got when we used our python script to calculate the surface interaction between each monomer. Chain A and C are the alpha subunits while chain B and D are the beta subunit.
So, we observe a strong interaction between subunit alpha and subunit beta.

Looking at this table, we observe that the major chains that interact are chain A with chain B and chain C with chain D.

Otherwise, the other monomers pairs show a bit smaller solvent accessible area, in particular considering chain A with chain D and chain B with chain C. Actually, considering the pair chain A and chain C, the surface interaction is much smaller in comparison with the other monomers pairs. To conclude, chain B and chain D have a so small interaction surface that their connection can be considered irrelevant with respect to the other monomer pairs.

### D. Relative Solvent Accessible Area Per Amino Acid

| Chain | Residue | RASA(M) | RASA(C) | RASA(M)-RASA(C) |
|---|---|---|---|---|
| A | GLU30 | 0.33 | 0.22 | 11.0% |
| A | ARG31 | 0.41 | 0.02 | 39.0% |
| A | LEU34 | 0.74 | 0.44 | 31.0% |
| A | SER35 | 0.7 | 0.14 | 57.0% |
| A | PRO37 | 0.54 | 0.3 | 24.0% |
| A | THR38 | 0.68 | 0.29 | 39.0% |
| A | LYS40 | 0.45 | 0.32 | 13.0% |
| A | THR41 | 0.74 | 0.05 | 69.0% |
| A | TYR42 | 0.32 | 0.11 | 21.0% |
| A | PRO44 | 0.72 | 0.46 | 26.0% |
| A | ARG92 | 0.74 | 0.18 | 56.0% |
| A | ASP94 | 0.57 | 0.06 | 51.0% |
| A | VAL96 | 0.58 | 0.32 | 26.0% |
| A | HIS103 | 0.51 | 0.09 | 42.0% |
| A | VAL107 | 0.32 | 0.0 | 32.0% |
| A | ALA110 | 0.3 | 0.0 | 30.0% |
| A | ALA111 | 0.57 | 0.09 | 48.0% |
| A | HIS112 | 0.31 | 0.17 | 14.0% |
| A | PRO114 | 0.69 | 0.44 | 25.0% |
| A | PHE117 | 0.2 | 0.01 | 18.0% |
| A | PRO119 | 0.77 | 0.15 | 61.0% |
| A | ALA120 | 0.49 | 0.29 | 20.0% |
| A | HIS122 | 0.44 | 0.01 | 43.0% |
| A | ALA123 | 0.49 | 0.16 | 33.0% |
| A | ASP126 | 0.52 | 0.13 | 39.0% |
| A | LYS127 | 0.42 | 0.12 | 31.0% |
| A | SER138 | 0.65 | 0.53 | 12.0% |
| A | TYR140 | 0.28 | 0.08 | 20.0% |
| A | ARG141 | 1.0 | 0.32 | 68.0% |

*Table 4: Relative Solvent Accessible Area table for chain A*

In these tables, we reported the residues that change the relative solvent accessible area when the pass from being alone in space to being in the haemoglobin tetramer.
Hence, looking at the last column, we selected only those amino acids whose RASA(Relative Accessible Salvent Area) changes particularly. This implies that these residues are important for the interactions of two monomers in the tetramer formation.
In particular, when want to highlight the hydrophobic spots which basically represent hydrophobic residues that feel a high loss of relative solvent accessible area when they took part of the tetramer. These spots probably represent areas of the surface of two monomers that interact through hydrophobic interactions. The number of hydrophobic hot spots that has been found differ per each chain: 14 for chain A, 16 for chain C, 13 for chain B and 13 for chain D.

In the table, we highlighted in gray the hydrophobic spots that we found. As we said before, these spots are the areas through which each monomer interact with the others in the tetramer organization. The tables reporting the relative accessible area of the other monomers are available in the supplementary material.

### E. Protein-Protein interaction netwrok table

The following tables report a part the proteins that directly interact with subunit alpha or subunit beta of the human haemoglobin protein.

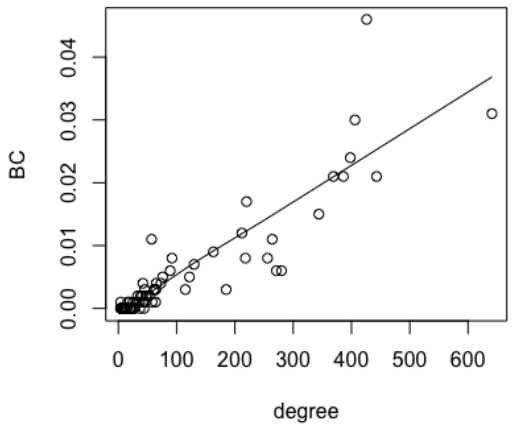
| Protein | Degree | C.coeff | BC |
|---|---|---|---|
| P69905 | 57 | 0.031 | 0.011 |
| P68871 | 42 | 0.031 | 0.004 |
| P19320 | 641 | 0.016 | 0.031 |
| P46013 | 443 | 0.02 | 0.021 |
| P11142 | 426 | 0.019 | 0.046 |
| Q16659 | 406 | 0.012 | 0.03 |
| Q15323 | 398 | 0.015 | 0.024 |
| Q6A162 | 386 | 0.011 | 0.021 |
| Q7Z3S9 | 369 | 0.035 | 0.021 |
| P27348 | 344 | 0.041 | 0.015 |

*Table 5 reports the proteins that directly interact with subunit alpha or subunit beta*

After the analysis that we performed, we ended with a total number of 80 proteins directly interact with subunit alpha or subunit beta within the human interactome. Per each protein we reported the degree, betweenness centrality and the clustering degree that they have in the subnetwork involving subunit alpha and subunit beta. In the table that we are showing, we just reported the two subunit alpha and beta (first and second row) and 8 from the 80 proteins that our computations provided. We listed them in numerical degree-descendent order. We highlighted also these values for the two subunits: the subunit alpha had a degree of 57 a clustering coefficient of 0.031 and a betweenness centrality of 0.011; Otherwise, the subunit beta had a degree of 42 a clustering coefficient of 0.031 and betweenness centrality of 0.004.

Then, looking at the table, we thought if there was some kind of correlation between some of these quantities. We observed that there was some level of positive correlation between the degree and the betweenness centrality of the nodes within the subgraph we extracted from the human proteome. As a consequence we calculated the Pearson Correlation coefficient to test the presence of this association.
To start, we plotted data:

**Figure 6: Plot of the degree and betweenness centrality**




The result provided a correlation coefficient of 0.9022321 and a P-Value of 2.2e-16, indicating the presence of positive correlation.

## 4. DISCUSSION

To conclude the analysis that we have conducted, we can highlight some aspects of the haemoglobin protein that we have understood starting from the results that we obtained.
In first place we can say that, during the formation of the tetramer, it is more likely that subunit alpha and subunit beta interact in a binary was. Basically, looking at the solvent accessible area, we see strong interaction between chain A (subunit alpha) and chain B (subunit beta). The same is observed for chain C (subunit alpha) and chain D (subunit beta).
The connections are guaranteed by hydrophobic interactions between hydrophobic amino acids and the overall structure stability is achieved by the formation of possible hydrogen bonds and salt bridges. In second place, looking at the interactions between the heteroatoms and the atoms of the monomers, the stability of the interaction of the haem group and oxygen in each monomer is related to the presence of histidine amino acids that interact with both the haem groups and the oxygen possibly through hydrogen bonds.
In third place, we can observe that, due to their high importance, both subunit alpha and subunit beta are highly connected to the other proteins since we observe an highly dense subnetwork from the human proteome where the the subunits are the main players.

## REFERENCES


1. **GeneCards:The Human Gene Database.**
   Database developed by the department of molecular genetics at the Weizmann Institut of science in Israel
   Web page link: https://www.genecards.org
2. **UCSC Genome Browser. University of Santa Cruz, California United States**
   Kent WJ, Sugnet CW, Furey TS, Roskin KM, Pringle TH, Zahler AM, Haussler D. The human genome browser at UCSC. Genome Res. 2002 Jun;12(6):996-1006.
   Web page link: https://genome.ucsc.edu
3. **NCBI:hemoglobin subunit alpha [Homo sapiens]**
   National Center for Biotechnology Information (NCBI). Bethesda (MD): National Library of Medicine (US).
   Web page link: https://www.ncbi.nlm.nih.gov/protein/NP_000508.1
4. **NCBI:hemoglobin subunit beta [Homo sapiens]**
   National Center for Biotechnology Information (NCBI). Bethesda (MD): National Library of Medicine (US).
   Web page link: https://www.ncbi.nlm.nih.gov/gene/3043
5. **Wikipedia:Hemoglobin**
   Wikipedia contributors. "Hemoglobin." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia.
   Web page link: https://en.wikipedia.org/wiki/Hemoglobin#Oxyhemoglobin
6. **oxy T state haemoglobin: oxygen bound at all four haems**
   Paoli, M., Liddington, R., Tame, J., Wilkinson, A., Dodson, G. (1996) J.Mol.Biol. 256: 775
   Web page link: https://www.rcsb.org/structure/1gzx
7. **Teach Me Physiology:Oxygen Transport in The Blood**
   Original Author(s): Will Clay. Last updated: 24th May 2018.
   Web page link: https://teachmephysiology.com/respiratory-system/transport-in-the-blood/oxygen-transport/
8. **Chemistry For Biologists**
   Author: Royal Society of Chemistry. Last updated November 2004.
   Web page link: https://www.rsc.org/Education/Teachers/Resources/cfb/transport.htm
9. **Protopedia-Hydrogen Bond**
   Prilusky J, Hodis E, Canner D, Decatur W, Oberholser K, Martz E, Berchanski A, Harel M, Sussman JL. Proteopedia: A status report on the collaborative, 3D web-encyclopedia of proteins and other biomolecules. J Struct Biol. 2011 Apr 23.
   Web page link: https://proteopedia.org/wiki/index.php/Hydrogen_bonds
10. **Protopedia-Salt Bridges**
    Prilusky J, Hodis E, Canner D, Decatur W, Oberholser K, Martz E, Berchanski A, Harel M, Sussman JL. Proteopedia: A status report on the collaborative, 3D web-encyclopedia of proteins and other biomolecules. J Struct Biol. 2011 Apr 23.
    Web page link: https://proteopedia.org/wiki/index.php/Salt_bridges
11. **Explaination DSSP Program**
    A series of PDB related databases for everyday needs. Wouter G Touw, Coos Baakman, Jon Black, Tim AH te Beek, E Krieger, Robbie P Joosten, Gert Vriend. Nucleic Acids Research 2015 January; 43(Database issue): D364-D368.
    Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Kabsch W, Sander C.
    Web page link: https://swift.cmbi.umcn.nl/gv/dssp/DSSP_3.html
12. **UCSF Chimera**
    Molecular graphics and analyses performed with UCSF Chimera, developed by the Resource for Biocomputing, Visualization, and Informatics at the University of California, San Francisco, with support from NIH P41-GM103311.
    UCSF Chimera–a visualization system for exploratory research and analysis. Pettersen EF, Goddard TD, Huang CC, Couch GS, Greenblatt DM, Meng EC, Ferrin TE. J Comput Chem. 2004 Oct;25(13):1605-12.
    Web page link: https://www.cgl.ucsf.edu/chimera/
13. **Intact EMBL-EBI protein-protein interaction database**
    Orchard S, Ammari M, Aranda B, Breuza L, Briganti L, Broackes-Carter F, Campbell NH, Chavali G, Chen C, del-Toro N, Duesbury M, Dumousseau M, Galeota E, Hinz U, Iannuccelli M, Jagannathan S, Jimenez R, Khadake J, Lagreid A, Licata L, Lovering RC, Meldal B, Melidoni AN, Milagros M, Peluso D, Perfetto L, Porras P, Raghunath A, Ricard-Blum S, Roechert B, Stutz A, Tognolli M, van Roey K, Cesareni G, Hermjakob H. The MIntAct project–IntAct as a common curation platform for 11 molecular interaction databases. Nucleic Acids Res. 2014 Jan;42(Database issue) D358-63. doi:10.1093/nar/gkt1115. PMID: 24234451; PMCID: PMC3965093.
    Web page link: https://www.ebi.ac.uk/intact/
14. **Uniprot: protein database**
    Morgat A, Lombardot T, Coudert E, Axelsen K, Neto TB, Gehant S,

Bansal P, Bolleman J, Gasteiger E, de Castro E, Baratin D, Pozzato M, Xenarios I, Poux S, Redaschi N, Bridge A, UniProt Consortium.
Web page link: https://www.uniprot.org

15. **Python package NetworkX**
Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, "Exploring network structure, dynamics, and function using NetworkX", in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008.
Web page link: https://networkx.github.io/documentation/stable/

16. **Numpy python module**
Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation, Computing in Science Engineering, 13, 22-30 (2011), DOI:10.1109/MCSE.2011.37.
Web page link: https://numpy.org/