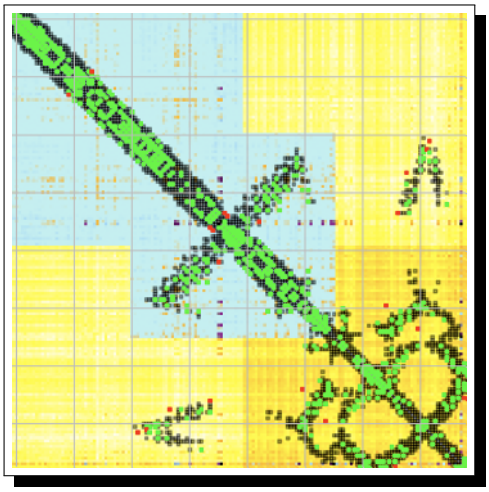
EPFL

MASTER THESIS

Machine-learning exploration of the interactions between chaperones and co-chaperones

Author:
Zoé MAJEUX

Professor:
Paolo De Los Rios



A thesis on molecular contacts predictions by direct coupling analysis and pseudolikelihood maximisation

Laboratory of Statistical Biophysics (LSB)

June 21, 2024

Abstract

Machine-learning exploration of the interactions between chaperones and co-chaperones

by Zoé MAJEUX

Proteins are the main characters of life and their functions are directly related to their structures that follow specific folding of blocs called amino acids. The folding process is mediated by a precise cooperation between proteins named chaperons and co-chaperons. The dysfunctionment of this co-working can lead to mis-formation and aggregation of incorrect proteins, resulting in neuromuscular and neurodegenerative disease, or lysosomal dyfunction. The understanding of chaperons and co-chaperons binding is a critical scientific interest to improve public health and could conduct to a better comprehension of disease as Alzheimer or Parkinson. Until today, structures of proteins can be obtain experimentally or with a computational tool referred as Alphafold. However, laboratory experiences take colossal time and sometimes are impossible for some sequences, and alphafold is a supervised learning model that was trained on structures. Here, a new model that requests only the proteins sequences for the training is presented and its application to different chaperons, co-chaperons and together are analysed. This one opens another vision with a direct couplings analysis approach and pseudolikelihood maximisation.

Introduction

The proteins and their evolution through species Involved in all processes of life, the proteins can be found in various shape or size but have a common attribute: they are composed of 20 different building blocks, with different chemical and electrical proprieties, linked together, and called amino acids (see Fig.1). These elements are the essence of life [1] and their origin is thought to have started with eleven standard amino acids in a prebiotic soup of chemical after the birth of the earth [2]. Today, 3.8 billion years after, computing the precise number of different combination of these blocks is as complicated to predict the exact number of different stars in the universe. However according to certain approximation there can have more than 50 billion of distinct proteins [3]. Since the specific roles of proteins are determined by the information encoded in their amino acid sequence [4], different proteins have similar regions if they accomplish similar task or due to evolutionary origin [5]. Moreover, the evolution has not only lead to similar proteins in the same organism but also across different taxonomies. Additionally, a reversible process, named phosphorylation/dephosphorylation, can result in alteration in the sequence of a protein [6]. This phenomena adds/removes a phosphate group (PO_4) to the polar group R of an amino acid resulting in a polarity change and allowing for example the interaction with other proteins [6]. All of this leads to a set of different proteins that are still similar and that are called homologous sequences.

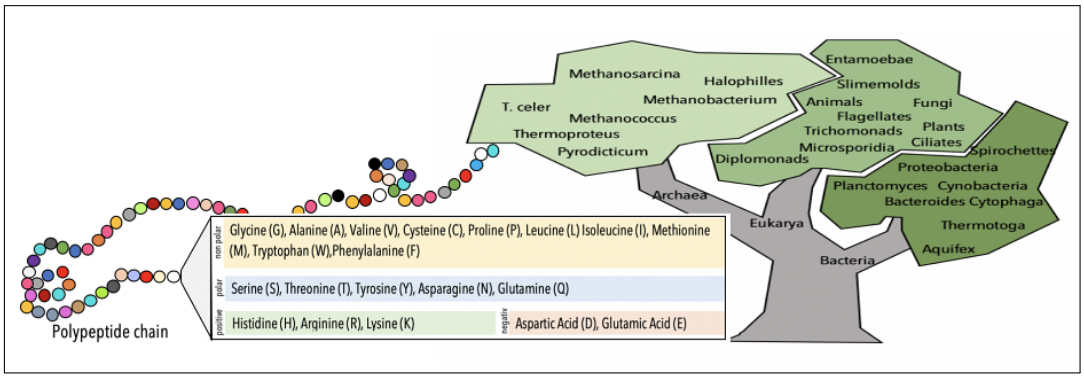


FIGURE 1: A protein is composed of 20 different amino acids with different polarities such that the evolution made this arrangement to be differentiate between organisms [7]

The chaperones and their different families To be able to exercise their precise roles, the proteins require a precise 3D conformation by passing through several steps: transcription of the DNA into RNA, translation of the RNA into protein and the folding of the sequence into its final shape. Certain proteins, known as chaperones, mediate and facilitate the folding of other proteins by binding to them and

stabilizing unfolded or partially folded polypeptide chains [8]. Without these chaperones, such chains may become unstable and aggregate with others. Many chaperones were initially called in 1975 Heat Shock Proteins (Hsp) [9] and initially known to facilitate the refolding of proteins that have been denatured with high temperature [8]. One prominent family member of these proteins is the Hsp70 that uses ATPase activity to plays important roles in various proteins processes including proteins folding, new proteins assembly, misfolded proteins refolding, and degradation of proteins [10]. Hsp70 constitutes an extremely conserved family of molecular chaperones in both prokaryotic and eukaryotic cells [11], and has been a high subject of interest for the scientific community for about fifty years [9]. Initial studies showcased its exceptional conservation across time and species [12]. The Hsp70s have a characteristic structure defined by three main parts: the N-terminal Nucleotide Binding Domain (NBD) that is connected by a short, highly conserved, and flexible linker to the Substrate Binding Domain (SBD), and followed by a disordered C-terminal portion [11]. Structural studies date back from the 90s, principally X-ray crystallography and Nuclear Magnetic Resonance, allowed to understand better their roles [13]. NBD was shown to have two lobes with a deep cleft and subdomains contributing to ATP binding and hydrolysis [14]. The SBD can clap the peptide chain thanks to its two parts: an α-helical lid domain and a β-sandwich domain (βSBD). The βSBD will enclose the peptide chain and the α-helical lid will restrict its access [13].

Moreover chaperone do not work alone but require binding with others particles to improve their activity: nucleotide exchange factors (NEFs) and J-domains proteins (JDPs) (see Fig.2). In 2005, scientists discovered that certain proteins, known as co-chaperones, could stimulate ATP activity to facilitate the folding function of Hsp70. DnaJ Proteins are a subset of these co-chaperones, and their association with Hsp70, which trigger the ATP hydrolysis to ADP, is critical to prevent the aggregation of non-native proteins [10]. On the other hand, NEFs play their critical role by inducing ADP dissociation allowing the binding of a new ATP molecule. The ATP binding will lead to the folded protein and NEF release [15].
Even if the Hsp70 family is distributed in every organisms, its proteins can differ between prokaryote or eukaryote. For prokaryote, Hsp70 are usually refereed to DnaK (still 50% identical to eukaryotic Hsp70 proteins) with a nucleotide exchange cofactor GroP-like E (GrpE) that will regulate the transformation of ADP to ATP [16]. GrpE are not only found in prokaryote but also in mitochondria and in chloroplasts of eukaryotic organism [17]. In eukaryote, an interesting chaperone of the family Hsp70 is the immunoglobulin heavy-chain binding protein (BiP). This chaperone plays an important role for eukaryote during the formation and folding of the proteins in the endoplasmic reticulum (ER) assuring only properly matured proteins to be transported outwhere [18]. The efficiency of this protein is improved with the nucleotide exchange factors Hyou1 (Grp170) [19]. The expression of BiP has been shown to be induced as the unfolded proteins in the ER are accumalated and to favorise the folding by preventing the aggregation [20].

The methods to understand the interaction between chaperones The interaction between chaperones and co-chaperones is of current scientific interest, since the correct functioning of our cells depends on it's correct pairing; leading to disease, or worst death, if in opposite it is not satisfied. Two principal consequences of a

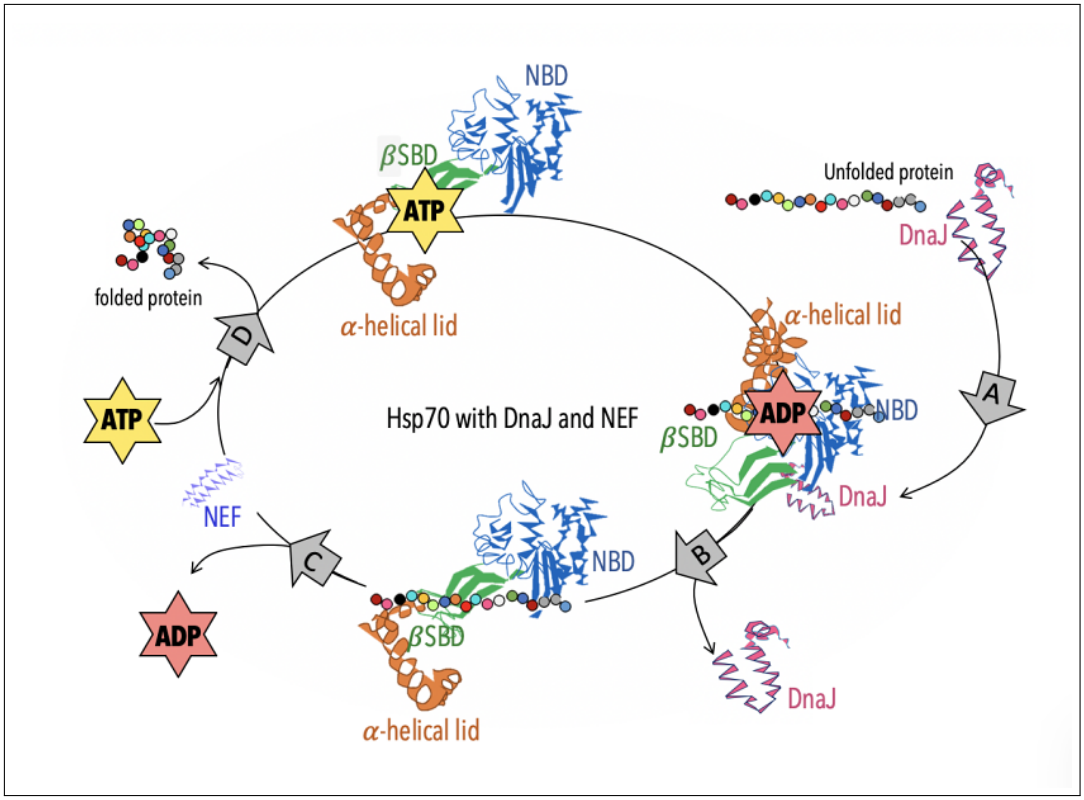


FIGURE 2: (A-B) the JDP binds to Hsp70, triggers the ATP hydrolysis to ADP, and then is released, (C-D) NEF induces ADP dissociation and the binding of a new ATP before to release the folded protein [7]

bad chaperone functioning are the Alzheimer disease and Parkinson disease that results from misfolded proteins that aggregate together [21]. The understanding of this interaction requests two mains capacities: (1) be able to assign correct pairs through similar sequences of identical family (GrpE, DnaK, BiP, Hyou1, ...) and identical organism, (2) be able to predict the binding contacts between these pairs. For now the best accurate prediction of binding contacts between a chaperone and a co-chaperone is obtained with AlphaFold that has been recognised as a solution to the protein folding problem by CASP14 benchmark[22]. This tool works as a supervised learning neural network taught with about 180'000 sequences and structures extracted experimental [22]. However, recently we developed and improve a new method able to detect coevolving pairs of amino acids in a single protein thanks to evolutionary information between homologous sequences and without any structural information [23]. This code[1] is based on direct couplings analysis with multiple sequence alignment and pseudolikelihood maximisation, and has been shown to predict similar contacts predictions than Alphafold for three specific sequences: the SIS1 protein (organism Saccharomyces cerevisiae) a Hsp40 co-chaperone containing DnaJ, the Mitochondrial protein import protein MAS5 (organism Saccharomyces cerevisiae) another type of Hsp40 co-chaperone containing DnaJ, and only DnaJ sequence (organism Oryza nivara). Additionally, previous analysis [24] showed the efficiency of the taxonomy information during the model learning. This was done by considering the taxonomy as a new amino acid of the sequence.

[1] https://github.com/zoemaj/DCA

This gives new perspectives in the architecture protein predictions that don't request previous structure information. Additionally, it can be used for other thematic (see Fig.3) as the phosphorylation or the taxonomic lineage. The model predictions can be used to reveal the two most probable amino acids per position allowing the detection of a tendency toward a polarity change for a specific amino position. Moreover, the taxonomic information in the learning process should allow the model to find this taxonomy for a specific sequence of amino acids.

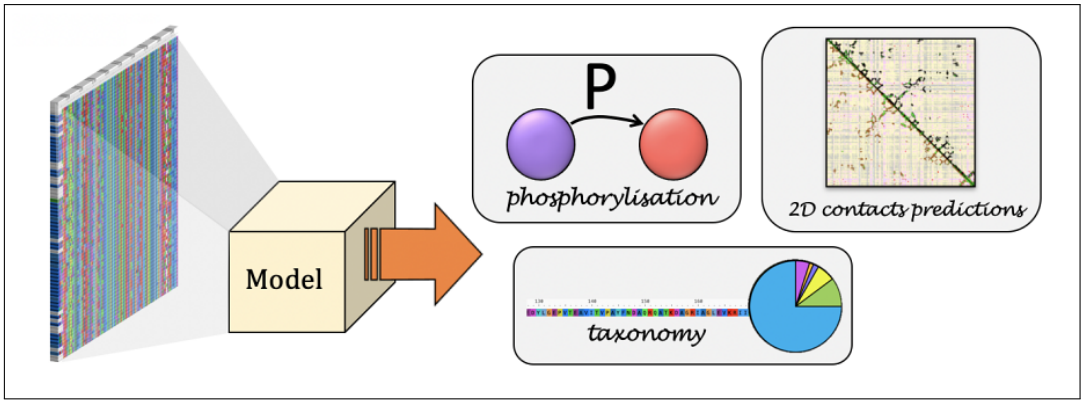


FIGURE 3: Different applications that the new model presented here that is based on based on direct couplings analysis with multiple sequence alignment and pseudolikelihood maximisation [7]

Here, a new version of this code is presented with two new models types and an improvement of contacts predictions by errors propagation. The contacts predictions between chaperone and co-factors are analysed with two different way. The first option considers the two proteins as only one single protein and applies the linear model on it. The second option tends to predict each proteins only by using the amino acids of the other protein and to use these relations to extract their contacts. This last option will determine if the information of the co-chaperone has been hided inside the chaperone (and inversely) thanks to the evolution. Besides, new optimisation analysis are going to be made on the optimiser parameters and the preprocessing of the data. In total five different types of sequences are studied: DnaJ, GrpE, DnaK, Hyou1 and BiP (see Fig.Fig.4). Each of this dataset will show unique results depending of several factors including the taxonomy levels, the dataset size and the numbers of amino acids.

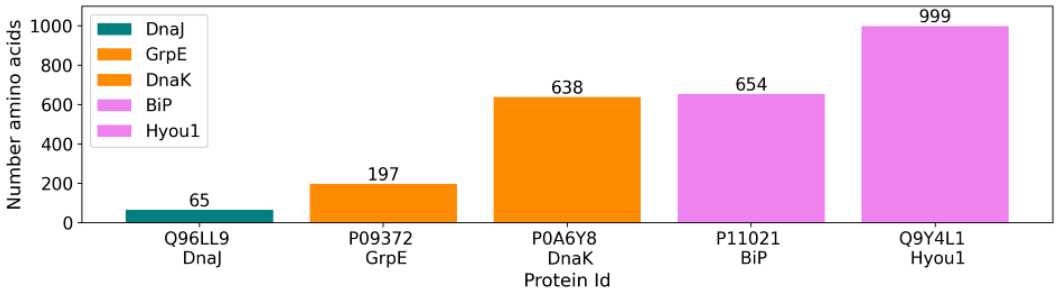


FIGURE 4: Proteins with their number of amino acids

Contents

List of Figures

Chapter 1

Statistical Theory

1.1 Direct Couplings Analysis (DCA)

1.1.1 Multiple sequence alignment (MSA)

Over the course of evolution, protein sequences can undergo modifications, leading to differences in the same protein among various species. However, despite these variations, the fundamental functions and three-dimensional structures of proteins should not be lost [27]. Proteins sharing such similarities are referred as homologous proteins and can be categorized into protein families through multiple sequence alignments (MSA) [27]. An example of homologous proteins is depicted in Figure 1.1, visualized using AliView [25].

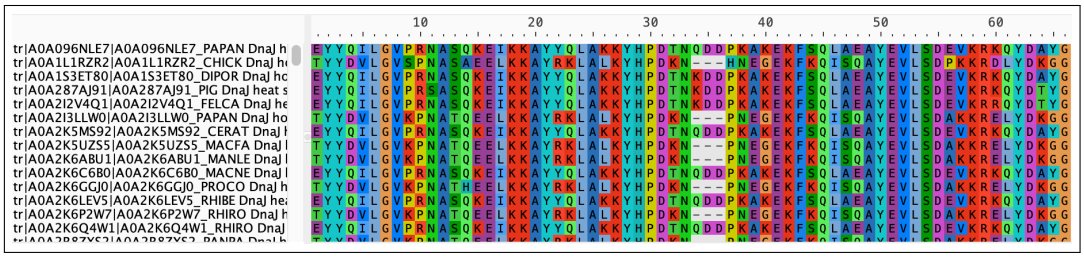


FIGURE 1.1: Crop of Aliview's results of some homologous sequences for DnaJ protein (IPR001623) [25]

MSA is used to represent each amino acids of L sequences of size N. This is represented by a rectangular matrix $A = \{(a_m)^l | i = 1, ...M, l = 1, ..., L\}$ containing L sequences (the rows) of size M. The columns m correspond to the amino acids at position m of the sequence $l \in [1, L]$ [24].

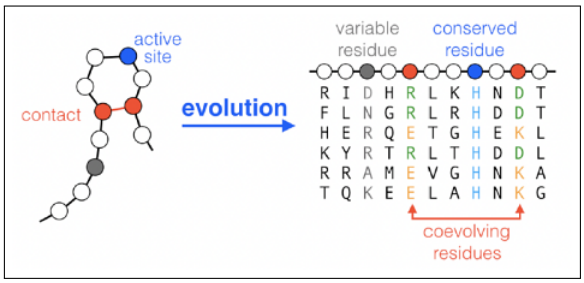


FIGURE 1.2: Information behind a MSA matrix [26]

The amino acid letter is replaced by a number $k \in [1, K]$ (in general K=21 because 20 natural amino acids, 1 gap). This matrix keeps a lot of secrets about the protein structure as the different contacts or correlations between the amino acids of a protein belonging to this family. Indeed, two amino acids in contact should mutate together as illustrated on Fig.1.2. This periodicity can be exploited by statistical models as direct coupling analysis (DCA) method that aims in providing a protein family-specific probability distribution by using the MSA (inverse statistical physics) and Potts model.

For the next formulations, these following notations are used:

- a_i: the amino acid at position i (i.e column i of the MSA)
- $(a_i)^l$: the value of the amino acid at position i and from the sequence l (i.e column i and row l of the MSA)
- $(a_i^\kappa)^l$: the amino acid at position i, of value k, and from the sequence l
- $\mathbf{a}$: the sequence of amino acids
$$(a_0, a_1, .., a_{M-1}, a_M)$$
- $\mathbf{a}_{/j}$: the sequence of amino acids without the jth element
$$(a_0, .., a_{i-1}, a_{i+1}, .., a_M)$$
- $C_{ij}^{\alpha\beta}$: the couplings between the amino acid at position i of value α and the amino acid at position j of value β

with $i, j \in [1, M], \alpha, \beta \in [1, K], l \in [1, L]$, M the length of the homologous sequences, K the number of different possible amino acid values, and L the number of sequences

1.1.2 Probability of Boltzman

First, note that given a MSA, the individual frequency, to have the amino acid β for the column j, and the pairwise frequencies, to have the pair (β, α) at columns j and i, are given by [24]:

$$\mathbf{f_j}(\beta) = \frac{1}{L}\sum_{l=1}^{L}\delta[(a_j)^l = \beta]$$

$$\mathbf{f_{j,i}}(\beta, \alpha) = \frac{1}{L}\sum_{l=1}^{L}\delta[(a_j)^l = \beta]\delta[(a_i)^l = \alpha]$$

with δ the Kronecker delta. These values are essential to predict the correlation $C_{ji}^{\beta\alpha}$:

$$C_{ji}^{\beta\alpha} = \mathbf{f_{j,i}}(\beta, \alpha) - \mathbf{f_j}(\beta)\mathbf{f_i}(\alpha)$$

Secondly, note that according to the Potts model, the probability to have the amino acid β at position j and the probability to have the couple (β, α) at position (j, i) (and the others amino acid of the sequences $\mathbf{a}_{/i}$ and $\mathbf{a}_{j,i}$ free) are given by the sum of probabilities of having each possible sequence with these values [24]:

$$P(a_j^\beta) = \sum_{\mathbf{a}_{/j}, a_j^\beta} P(\mathbf{a}) = \mathbf{f_j}(\beta)$$

$$P(a_j^\beta, a_i^\alpha) = \sum_{\mathbf{a}_{/j,i}, a_j^\beta, a_i^\alpha} P(\mathbf{a}) = \mathbf{f_{j,i}}(\beta, \alpha)$$

Assuming the MSA to be a sample of a Boltzmann distribution, the probability to find the sequence $\mathbf{a} = (\mathbf{a})^l$ is given by $P(\mathbf{a}) = \frac{1}{Z}\exp(-\beta\mathbf{H})$ with $\mathbf{H}$ a matrix of size

MxK representing the field of each amino $(a_m^\kappa)^l$. By taking the maximization of the entropy $S = -\sum_{\mathbf{a}} P(\mathbf{a}) \ln P(\mathbf{a})$ the probability becomes :

$$P(\mathbf{a}) = \frac{1}{Z} \exp\left(\sum_{m=1}^{M} \mathbf{h_m}(a_m) + \sum_{m=1}^{M-1} \sum_{i=m+1}^{M} \mathbf{J_{mn}}(a_m, a_n) \right) \tag{1.1}$$

with Z a normalization constant, $\mathbf{h_m} = (h_{m,1}, .., h_{m,K})^T$ a vector of fields and $\mathbf{J_{mn}} = \begin{pmatrix} C_{mn,11} & ... & C_{mn,1K} \\ ... & ... & ... \\ C_{mn,K1} & ... & C_{mn,KK} \end{pmatrix}$ the matrix of couplings between the positions $(a_m)^l$ and $(a_n)^l$ for different values of amino acids $\in [1, K]$ [24]. A large H_m^κ indicates a preference of position m toward the amino acid k and a large $\mathbf{J_{mn,kp}}$ indicates a high contact probability between the positions (m, n) of amino acids values (k, p) [24]. The inferring problem consists in finding these quantities.

1.1.3 Alternative representation of MSA in one hot encoder

The MSA gives a rectangular matrix $A = \{(a_j)^l | j = 1, ...M, l = 1, ..., L\}$ such that each amino acid $(a_j)^l$ of the sequence l takes a value $k \in [1, K]$. This matrix can be expressed as a 3 dimensional tensor of size $LxMxK$ such that each amino acid $(a_j)^l$ is now a one hot encoder vector of dimension K (see Fig.1.3). This representation gives the following condition:

$$\sum_{\kappa=1}^{K} (a_j^\kappa)^l = 1 \qquad \forall l \in [1, L] \text{ and } \forall j \in [1, M] \tag{1.2}$$

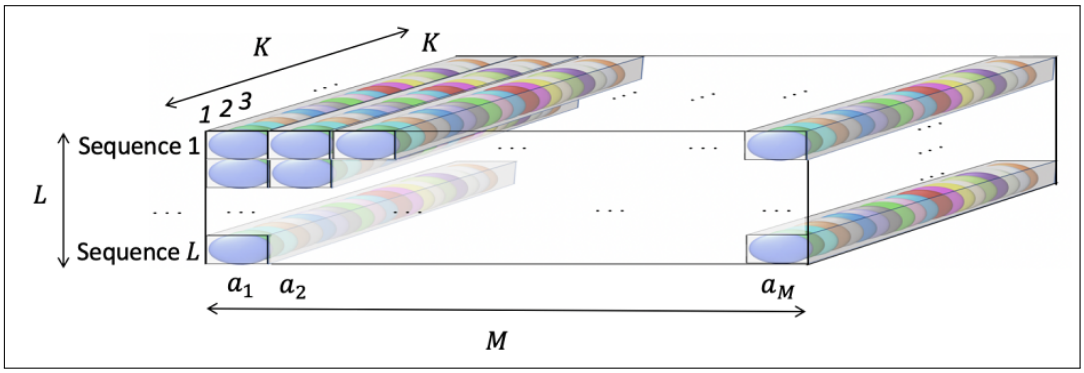


FIGURE 1.3: Multiple sequence alignments of L sequences with M amino acids of values K [7]

1.2 Inferring with pseudolikelihood maximisation

1.2.1 Pseudolikelihood

The pseudo likelihood to find the sequence of amino acid **a** is given by the product of the probabilities of each amino acid j knowing the probability of the others .

$$P(\mathbf{a}) = \prod_{j}^{M} P(a_j|\mathbf{a}_{/j}) \tag{1.3}$$

In the MSA there are L sequences, thus the probability to have the amino acid j is given by the product along each sequence of each probability to have the amino acid j knowing the other amino acids of the sequences

$$P(a_j|\mathbf{a}_{/j}) \approx \prod_{l=1}^{L} P(a_j = a_j^l|\mathbf{a}_{/j} = \mathbf{a}_{/j}^l) \tag{1.4}$$

By using the Bayes theorem $P(A|B) = \frac{P(A\cap B)}{P(B)}$ [28], the probability to have the amino acid a_j in the sequence l knowing the others amino acids of the sequence is given by:

$$P(a_j = (a_j)^l|\mathbf{a}_{/j} = (\mathbf{a}_{/j})^l) = \frac{P(a_j = (a_j)^l, \mathbf{a}_{/j} = (\mathbf{a}_{/j})^l)}{P(\mathbf{a}_{/j} = (\mathbf{a}_{/j})^l)} \overset{(*^1)}{=} \frac{P(a_j = (a_j)^l, \mathbf{a}_{/j} = (\mathbf{a}_{/j})^l)}{\sum_{\beta=1}^{\kappa} P(a_j^\beta, (\mathbf{a}_{/j})^l)}$$
$$\underline{\underline{(*^2) \text{ and } (*^3)}} \frac{\exp\left(\mathbf{h_j}(a_j)^l + \sum_{i\neq j}^{M} \mathbf{J_{ji}}((a_j)^l, (a_i)^l)\right)}{\sum_{\beta=1}^{K} \exp\left(\mathbf{h_j}(\beta) + \sum_{i\neq j}^{M} \mathbf{J}_{ji}(\beta, (a_i)^l)\right)}$$

Using the representation in one hot shot encoder for the MSA the probability from Eq.(1.4) is now $P(a_j^\beta)$ to have the amino acid β at position j and is given by :

$$P(a_j|\mathbf{a}_{/j}) \to P(a_j^\beta|\mathbf{a}_{/j}) \approx \prod_{l=1}^{L} P(a_j^\beta = (a_j^\beta)^l|\mathbf{a}_{/j} = (\mathbf{a}_{/j}^l) \tag{1.5}$$

By using some proprieties with $\mathbf{h_j}$ and $\mathbf{J_{ji}}$ $(*^4)$, the probability to find a_j^b in the sequence l can be written as:

$$P(a_j^\beta = (a_j^\beta)^l|\mathbf{a}_{/j}^\beta = (\mathbf{a}_{/j}^\beta)^l) = \frac{\exp\left(\sum_{\kappa}^{K} H_j^\kappa \cdot (a_j^\kappa)^l + \sum_{i\neq j}^{M}\sum_{\kappa,\phi}^{K} C_{ji}^{\kappa\phi} \cdot (a_j^\kappa)^l \cdot (a_i^\phi)^l\right)}{\sum_{\beta=1}^{K} \exp\left(H_j^\beta + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\beta\phi} \cdot (a_i^\phi)^l\right)}$$

And the probability to find the amino acid a_j^β in the MSA in Eq.(1.5) becomes:

$$P(a_j^\beta|\mathbf{a}_{/j}) \approx \prod_{l=1}^{L} \frac{\exp\left(\sum_{\kappa}^{K}\left[(a_j^\kappa)^l \cdot (H_j^\beta + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\kappa\phi} \cdot (a_i^\phi)^l)\right]\right)}{\sum_{\beta=1}^{K} \exp\left(H_j^\beta + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\beta\phi} \cdot (a_i^\phi)^l\right)} \tag{1.6}$$

The computation details $*^1$, $*^2$, $*^3$ and $*^4$ are found in Appendix.A

The final pseudo-likelihood given in Eq.(1.3) becomes with Eq.(1.6):

$$\boxed{P(\mathbf{a}) = \prod_{j=1}^{M}\prod_{l=1}^{L} \frac{\exp\left(\sum_{\kappa}^{K}\left[(a_j^{\kappa})^l \cdot (H_j^{\beta} + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\kappa\phi}\cdot(a_i^{\phi})^l)\right]\right)}{\sum_{\beta=1}^{K}\exp\left(H_j^{\beta} + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\beta\phi}\cdot(a_i^{\phi})^l\right)}} \tag{1.7}$$

1.2.2 Maximisation of pseudolikelihood

To avoid the maximisation on the double product (and transform them into summations), the maximisation will be done on it's logarithm. Since the logarithmic function is increasing, the maximum of a function is equivalent to the maximum of the logarithm of this function. Additionally, the maximum of a function is equivalent to the minimum of its negation, and the multiplication of a constant doesn't change the problem. These managements are a common approach during inferring problem for computation cost reduction:

$$\boxed{\text{Max}(P(\mathbf{a})) \Leftrightarrow \text{Min}(-\frac{1}{M}\sum_{j}^{M}\log P(a_j^{\beta}|\mathbf{a}_{/j}))} \tag{1.8}$$

1.3 Cross entropy

The minimisation of $-\frac{1}{M}\sum_{j}^{M}\log P(a_j^{\beta}|\mathbf{a}_{/j})$ (see Eq.(1.8)) with $P(a_j^{\beta}|\mathbf{a}_{/j})$ given in Eq.(1.6) can be simplified by using the propriety of the normalisation of the one hot encoders amino acids mentioned in Eq. (1.2) .

$$\begin{aligned}\mathcal{L} &= -\frac{1}{M}\sum_{j=1}^{M}\log P((a_j^{\beta})^l) \overset{(1.6)}{=} -\frac{1}{M}\sum_{j=1}^{M}\sum_{l=1}^{L}\log\left[\frac{\exp\left(\sum_{\kappa}^{K}\left[(a_j^{\kappa})^l(H_j^{\kappa} + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\kappa\phi}\cdot(a_i^{\phi})^l)\right]\right)}{\sum_{\beta}^{K}\exp\left(H_j^{\beta} + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\beta\phi}\cdot(a_i^{\phi})^l\right)}\right] \\ &\overset{*^5}{=} -\frac{1}{M}\sum_{j=1}^{M}\sum_{l=1}^{L}\sum_{\kappa}^{K}(a_j^{\kappa})^l\log\left[\frac{\exp\left(\left[((H_j^{\kappa} + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\kappa\phi}\cdot(a_i^{\phi})^l)\right]\right)}{\sum_{\beta}^{K}\exp\left(H_j^{\beta} + \sum_{i\neq j}^{M}\sum_{\phi}^{K} C_{ji}^{\beta\phi}\cdot(a_i^{\phi})^l\right)}\right]\end{aligned}$$

Which can be expressed as:

$$\boxed{\mathcal{L} = -\frac{1}{M}\sum_{l,j,k}(a_j^k)^l\log\textbf{Softmax}\left((z_j^{\beta})^l\right)} \tag{1.9}$$

with $(z_j^{\beta})^l \in [1, K]$ the output predicted by the model:

$$\boxed{(z_j^{\beta})^l = H_j^{\kappa} + \sum_{i\neq j,\phi} C_{ji}^{\kappa\phi}\cdot(a_i^{\phi})^l} \tag{1.10}$$

The computation details $*^5$ are found in Appendix.A

such that the application of the Softmax at this output is given by:

$$\boxed{\textbf{Softmax}(z_j^\beta)^l = \frac{\exp{(z_j^\beta)^l}}{\sum_\beta^K \exp{(z_j^\beta)^l}} = (\hat{y}_j^\beta)^l} \tag{1.11}$$

This loss function corresponds to the cross entropy loss $\mathcal{L} = -\sum_a y_a \cdot \log{(\hat{y}_a)}$ known to measure the distance between the predictions of a model given by $\hat{y}_a$ and the truth given by y_a. In this case, $y_a = (a_j^k)^l$ is the value of the amino acid and $\hat{y}_a = \textbf{Softmax}((z_j^\beta)^l) = \textbf{Softmax}(H_j^\kappa + \sum_{i \neq j, \phi} C_{ji}^{\kappa\phi} \cdot (a_i^\phi)^l)$ is the predicted $(\hat{a}_i^k)^l$ value of the amino acid a_i of the sequence l

Chapter 2

Computation Concept

The computation process can be divided into seven steps belonging to four main parts that are illustrated on Fig.2.1. The first part is the treatment of the data and labels with alignment, preprocessing and sequence reweighting to have the more appropriates **a**. The second part is the model building and training to infer the values of $\hat{\mathbf{h}}$ and $\hat{\mathbf{J}}$. The third part is the extraction of the couplings **C** to have a contact map of size $M \times M$. And the fourth part handles the errors propagation and their application to the contacts map.

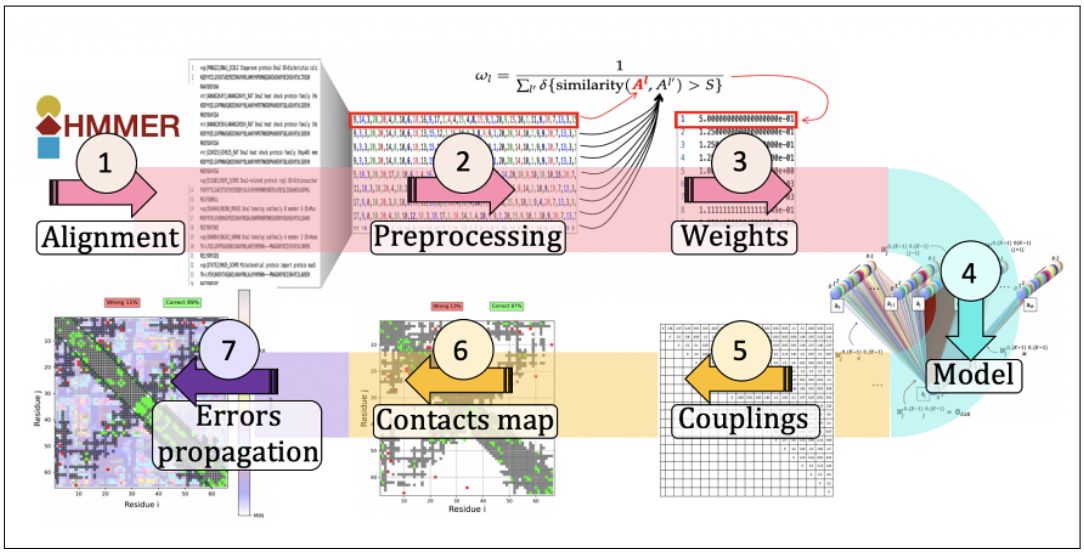


FIGURE 2.1: After the alignment of the sequences (1), a preprocessing filters the sequences according to their gaps and similarities with the first one (2). This is followed by weights attribution (3) that penalizes the most common sequences in the MSA. This is followed by the model building and training (4), and the extraction of the couplings (5). The related contacts score are visualize on a map (6). Finally, an improvement of the map can be done by considering the errors propagation (7). [7]

2.1 Data and labels

2.1.1 Extraction of the data

The data are extracted from the Universal Protein Resource (UniProt) [29] and aligned according to a reference sequence with the version 3.4 of Hidden Markov Model based tool (HMMER) [30]. For the working of this alignment, the extraction of

the profile HMM of the sequences family is necessary and is found with the large database of protein families groups PFAM [31] accessible by the bioinformatics resource Interpro [32].

2.1.2 Preprocessing

The first preprocessing consist in filtering the sequences according to their similarity with the sequence of reference l_{ref} used for the alignment (and that will be used for the contact map of reference). The remaining sequences l should respect the constraint given by the minimum m_{sim} and maximum M_{sim} of similarity accepted :

$$m_{\text{sim}} < \sum_i \delta\{a_i^l, a_i^{l_{\text{ref}}})\} < M_{\text{sim}}$$

The second preprocessing consists in filtering the sequences according to their amount of gaps (-) causing by the alignment. The remaining sequences l should respect the constraint given by the threshold amount X of gaps accepted:

$$\sum_i \delta\{a_i^l, -)\} < X$$

2.1.3 Varying number of K

A filtering is made according to the allowed k values per position. If the MSA doesn't contain a certain value k for $\boldsymbol{a}_i$, the model will not try to find it. For each amino position, the possible values k between 0 to 20 that are not present in the L amino acids are masked with all the amino acids a_j^β with $j \in [1, M]$ and $\beta \in [1, K]$.

2.1.4 Sequence reweighting

Let's remind the assumption that the MSA was a sample following a Boltzman distribution such that each sample were statistically independent. This is not the case with the data since their extraction could lead to a number of sequences that are highly similar (for example if a huge amount of sequences come from the same species). The inequality of similitude can be corrected by sequence re-weighting i.e by attributed lower weights for sequences more common and higher weights for sequences more rare. The similarity between two sequences can be calculated as $\text{similarity}(A^l, A^{l'}) = \sum_m \delta\{a_m^l = a_m^{l'}\}$. Let's fix a threshold of similarity S, then the weight of a sequence l is given by:

$$\omega_l = \frac{1}{\sum_{l'} \delta\{\text{similarity}(A^l, A^{l'}) > S\}}$$

And the loss from Eq.(1.9) becomes:

$$\mathcal{L} = -\frac{1}{M} \sum_{l,m,k} \frac{1}{\omega_l} a_{m,k}^l \log \left[\textbf{Softmax}(H_{m,k'} + \sum_{n \neq m,p} C_{m,k',n\neq m,p} a_{n,p}^l) \right] \tag{2.1}$$

2.2 Architectures of the model

The predicted $\hat{\mathbf{h}}$ and $\hat{\mathbf{J}}$ (given the outputs $(z_j^\beta)^l$ defined in Eq.1.10) are learned with one of the three following architectures : linear, linear with cross proteins, and no

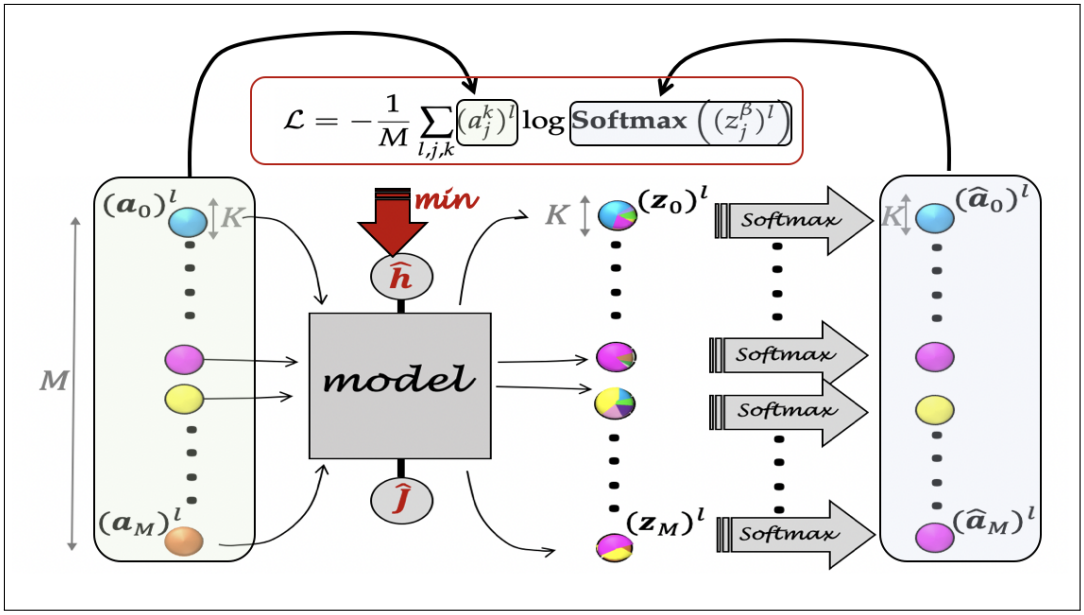


FIGURE 2.2: Labels predictions with one classifier l. MK outputs $(\mathbf{z})^l$ are computed by the model with M input $(\mathbf{a})^l$ before to extract the M predictions $(\hat{\mathbf{a}})^l$ by applying the Softmax one each bloc of size K. [7]

linear.

Considering the input and labels as same one hot encoders elements, each different architecture is composed of L (the number of sequences) different classifiers that have to predict the value of an amino acid as function of all the others amino acids (see Fig.2.2).

2.2.1 Linear

The Eq.(1.10) can be seen as a linear relation with:

Bias:	$\mathbf{W}_j^{(0)} = \mathbf{h}_j \in \mathbb{R}^K$	$\forall j \in [0, M]$
Non-null weights:	$\mathbf{W}_{\mathbf{ji}}^{(\mathbf{1})} = \mathbf{J}_{ji} \in \mathbb{R}^{KxK}$	$\forall j, i \in [0, M] \cap (j \neq i)$
Null weights:	$\mathbf{W}_{\mathbf{ji}}^{(\mathbf{1})} = \mathcal{O} \in \mathbb{R}^{KxK}$	$\forall j, i \in [0, M] \cap (j = i)$

From this architecture, the output of the amino acid at position j with value κ in the sequence l is referred as $(z_j(\kappa))^l$ and its expression is given by (see Fig.2.3 for a visual representation):

$$(z_j(\kappa))^l = W_j^{(0)}(\kappa) + \sum_{\phi}^{K}\sum_{i}^{M} W_{ji}^{(1)}(\kappa, \phi) \cdot (a_i^\phi)^l \tag{2.2}$$

From this relation it is possible to express for each classifier l only the dependence between the output $(z_j^{\alpha})^l$ and the input $(a_i^{\phi})^l$:

$$(z_j(\kappa))^l = W_j^{(0)}(\kappa) + \sum_{\phi}^{K} \left(W_{j1}^{(1)}(\kappa,\phi) \cdot (a_1^{\phi})^l + .. + W_{jM}^{(1)}(\kappa,\phi) \cdot (a_M^{\phi})^l \right) \tag{2.3}$$

$$(z_{ji}(\kappa,\phi))^l = W_{ji}^{(0)}(\kappa) + \sum_{\phi}^{K} W_{ji}^{(1)}(\kappa,\phi) \cdot (a_i^{\phi})^l \quad \text{with } W_{ji}^{(0)} = \frac{1}{M} W_j^{(0)} \tag{2.4}$$

From Eq.(2.2), the bias and the weights of the models can be extracted. By taking an input sequence such that $(a_i^{\phi})^l = \delta(i,i^{\star}) \cdot \delta(\phi,\phi^{\star})$ only one term of the double sum will be no zero and the weights $W_{ji^{\star}}^{(1)}(\kappa,\phi^{\star})$ can be found:

$$\begin{cases} W_{ji^{\star}}^{(1)}(\kappa,\phi^{\star}) = (z_j(\kappa))^l - W_j^{(0)}(\kappa) \in \mathbb{R}^{MxK} & \forall \kappa \in [1,K], \text{and } j \in [1,M] \quad (2.5a) \\ (a_i^{\phi})^l = \delta(i,i^{\star}) \cdot \delta(\phi,\phi^{\star}) & \forall \phi \in [1,K], \text{and } i \in [1,M] \quad (2.5b) \end{cases}$$

Every possible values of $\mathbf{W}^{(1)}$ can be extracted from the $M \cdot K$ inputs such that $(a_i^{\phi})^l = \delta(i,i^{\star}) \cdot \delta(\phi,\phi^{\star})$ with every possible configuration of pairs $(i^{\star},\phi^{\star})$.

By taking an input sequence such that $(a_i^{\phi})^l = 0 \; \forall i \in [1,M], \phi \in [1,K]$ only the bias term $W_j^{(0)}$ $(\forall j \in [1,M])$ and the output term $(z_j)^l$ remain in the Eq (2.2) and the bias can be extracted:

$$\begin{cases} W_j^{(0)}(\kappa) = (z_j(\kappa))^l & \forall \kappa \in [1,K], \text{and } j \in [1,M] \quad (2.6a) \\ (a_i^{\phi})^l = 0 & \forall \phi \in [1,K], \text{and } i \in [1,M] \quad (2.6b) \end{cases}$$

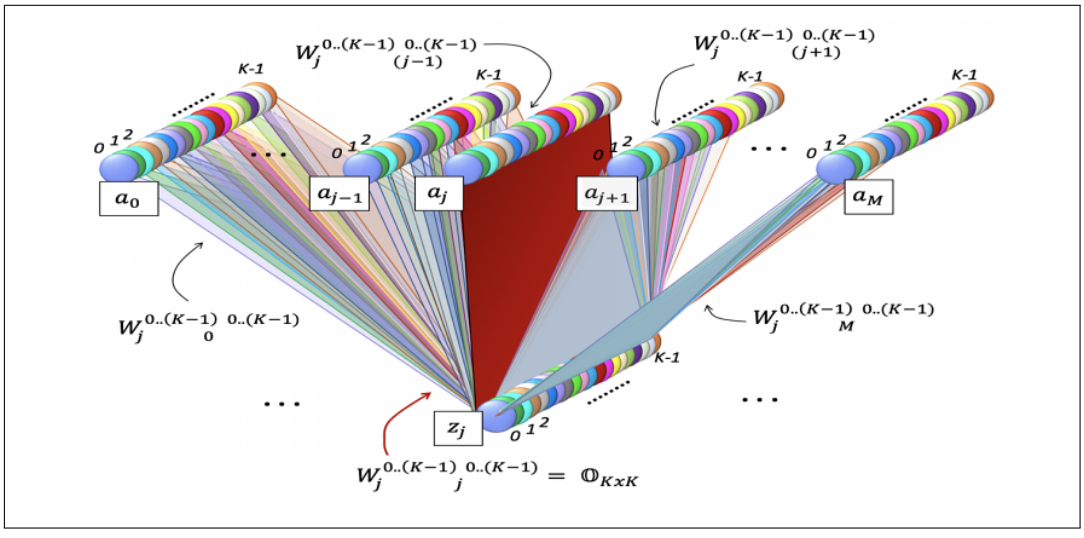


FIGURE 2.3: Linear weights between M inputs of size K and M outputs of size K. Masks are applied between every same position j to prevent self learning (red plan). [7]

2.2.2 Linear with cross proteins

Considering two proteins A and B with respectively M_A and M_B amino acids, it is possible to predict only the amino acids of a protein A with the amino acids of a protein B and vice versa. In this case the two proteins are concatenated together to form a new protein with $M = M_A + M_B$ amino acids, and the linear with cross proteins architecture is used to separate the information provenance (see Fig.2.4 for a visual interpretation). In this case, the Eq.(1.10) is seen as a linear relation with:

Bias:	$\mathbf{W}_j^{(0)} = \mathbf{h}_j \in \mathbb{R}^K$	$\forall j \in [0, M]$
Non-null weights for A:	$\mathbf{W}_{\mathbf{ji}}^{(1)} = \mathbf{J}_{ji} \in \mathbb{R}^{KxK}$	$\forall j \in [0, M_A], i \in [M_A, M_B]$
Non-null weights for B:	$\mathbf{W}_{\mathbf{ji}}^{(1)} = \mathbf{J}_{ji} \in \mathbb{R}^{KxK}$	$\forall j \in [M_A, M_B], i \in [0, M_A]$
Null weights for A:	$\mathbf{W}_{\mathbf{ji}}^{(1)} = \mathcal{O} \in \mathbb{R}^{KxK}$	$\forall j, i \in [0, M_A]$
Null weights for B:	$\mathbf{W}_{\mathbf{ji}}^{(1)} = \mathcal{O} \in \mathbb{R}^{KxK}$	$\forall j, i \in [M_A, M_B]$

The outputs, weights and bias are given respectively by Eq.(2.2), Eq.(2.5) and Eq.(2.6).

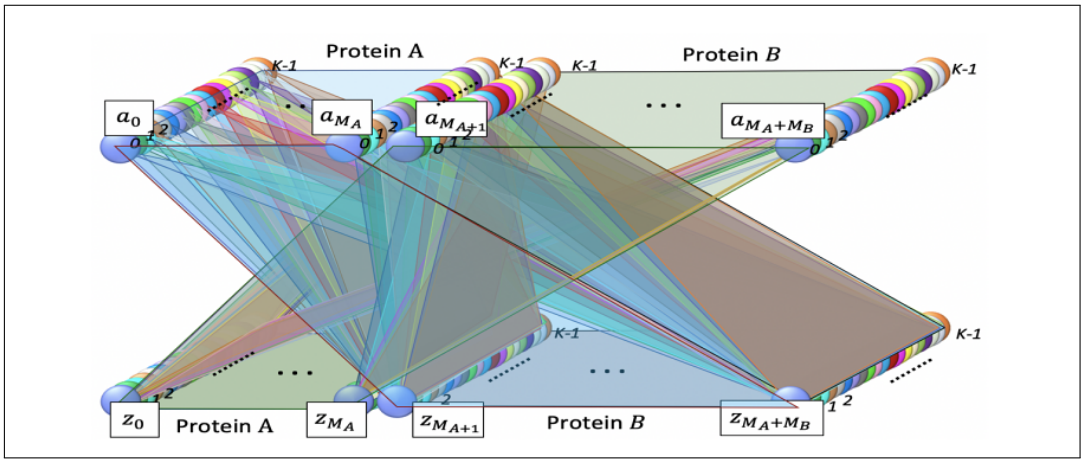


FIGURE 2.4: linear weights between M inputs of size K and M outputs of size K for two proteins. Masks are applied between every positions j belonging to the same protein. [7]

2.2.3 No linear

The Eq.(1.10) can be seen as a non-linear relation with:

Bias:	$\mathbf{W}_y^{(0)} = \mathbf{h}_y \in \mathbb{R}^K$	$\forall y \in [0, M]$
Non null hidden weights:	$\mathbf{W}_{\mathbf{ji}}^{(1)} = \mathbf{J}_{ji} \in \mathbb{R}^{KxK}$	$\forall j, i \in [0, M] \cap (j \neq i)$
Null hidden weights:	$\mathbf{W}_{\mathbf{ji}}^{(1)} = \mathcal{O} \in \mathbb{R}^{KxK}$	$\forall j, i \in [0, M] \cap (j = i)$
Non null weights:	$\mathbf{W}_{\mathbf{jyi}}^{(2)} = \mathbf{C}_{jyi} \in \mathbb{R}^{KxKxK}$	$\forall j, y, i \in [0, M] \cap (i = j) \cap (y \neq i)$
Null weights:	$\mathbf{W}_{\mathbf{jyi}}^{(2)} = \mathcal{O} \in \mathbb{R}^{KxKxK}$	$\forall j, y, i \in [0, M] \cap (j \neq i) \cap (y = i)$

From this architecture, the output can be written as:

$$(z_j(\kappa))^l = \left(W_j^{(0)}(\kappa) + \sum_{\tilde{\kappa}}^{K} W_j^{(1)}(\kappa, \tilde{\kappa}) \right) \cdot \mathcal{F}(b_j^{\tilde{\kappa}})^l \tag{2.7}$$

$$= \left(W_j^{(0)}(\kappa) + \sum_{\tilde{\kappa}}^{K} W_j^{(1)}(\kappa, \tilde{\kappa}) \right) \cdot \mathcal{F} \left(\tilde{W}_j^{(0)}(\tilde{\kappa}) + \sum_{\phi}^{K} \sum_{i}^{M} \tilde{W}_{ji}^{(1)}(\tilde{\kappa}, \phi) \cdot (a_i^{\phi})^l \right) \tag{2.8}$$

$$\overset{\text{taylor}}{=} W_j^{(0)}(\kappa) + \sum_{\phi}^{K} \sum_{i}^{M} W_{ji}^{(1)}(\kappa, \phi) \cdot (a_i^{\phi})^l + \sum_{\phi}^{K} \sum_{\gamma}^{K} \sum_{i}^{M} W_{jiy}^{(2)}(\kappa, \phi, \gamma) \cdot (a_i^{\phi})^l \cdot (a_y^{\gamma})^l + ... \tag{2.9}$$

From Eq.(2.9), the bias and the weights of the models can be extracted. The bias $\mathbf{W}^{(0)}$ and the hidden weights $\mathbf{W}^{(1)}$ can be extracted as defined in Eq.(2.6) and (2.5). Next By taking an input sequence such that $(a_i^{\phi})^l = \delta(i, i^\star) \cdot \delta(\phi, \phi^\star)$ and $(a_y^{\gamma})^l = \delta(y, y^\star) \cdot \delta(\gamma, \gamma^\star)$ with $i \neq y$ only one term of the triple sum will be no zero and the weights $W_{jiy}^{(2)}(\kappa, \phi^\star, \gamma^\star)$ can be found:

$$\begin{cases} W_{ji^\star y^\star}^{(2)}(\kappa, \phi^\star, \gamma^\star) = (z_j(\kappa))^l - W_j^{(0)}(\kappa) & \forall \kappa \in [1, K], j \in [1, M] \\ \qquad\qquad - W_{ji^\star}^{(1)}(\kappa, \phi^\star) \cdot (a_{i^\star}^{\phi^\star})^l - W_{jy^\star}^{(1)}(\kappa, \gamma^\star) \cdot (a_y^{\gamma^\star})^l & \\ (a_i^{\phi})^l = \delta(i, i^\star) \cdot \delta(\phi, \phi^\star) & \forall \phi \in [1, K], i \in [1, M]/y \\ (a_y^{\gamma})^l = \delta(y, y^\star) \cdot \delta(\gamma, \gamma^\star) & \forall \gamma \in [1, K], y \in [1, M]/i \end{cases}$$

the values of $\mathbf{W}^{(2)}$ can be extracted from the $\frac{MK \cdot (MK-1)}{2}$ inputs such that every possible configuration (without repetition) of $(i^\star, \phi^\star)$, $(y^\star, \gamma^\star)$ and $i \neq y$ are used.

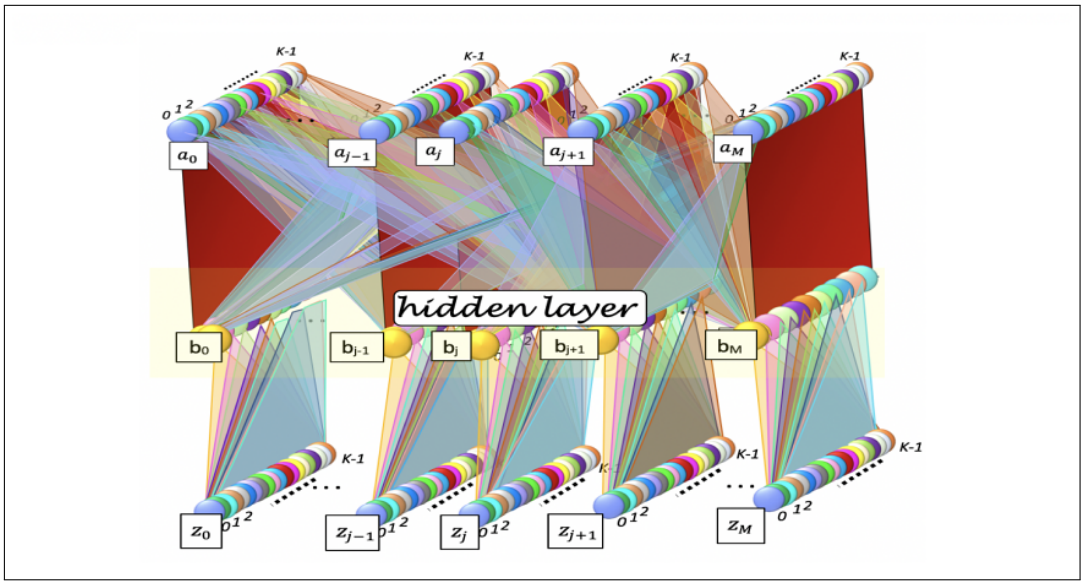


FIGURE 2.5: Non linear weights (induces by a hidden layer) between M inputs of size K and M outputs of size K. A first mask is applied between every same positions j of the first layer and the hidden layer. A second mask is applied between no similar positions i and j of the hidden layer and the last layer. [7]

2.2.4 Numbers of parameters

The model tends to minimise (Eq.(1.8)) a solution depending of a number of parameters $\mathcal{N}_{\text{param}}$. This number is computed according to the different architectures. However, it can be shown that this number is larger than the theoretical values which implies a model over-parametered and a model that cannot converge to a solution (different parameters set can describe the same probability distribution). Without a loss of generality, let's see the linear case: The number of parameters $N_{\text{param},j}$ that the model needs to learn to predict the output z_j is given by (according to (2.2)):

$$\begin{aligned}\mathcal{N}_{\text{param},j} &= \mathcal{N}_{W_j^{(0)}(\kappa)} + \mathcal{N}_{W_{ji}(\kappa,\phi)^{(1)}} \\ &= K + (M-1)K^2\end{aligned}$$

By considering that $W_{ji} = W_{ij}$ and that there are M different j, the number of parameters $\mathcal{N}_{param}$ that the model needs to learn to predict every output is given by:

$$\mathcal{N}_{param} = MK + \frac{M(M-1)}{2}K^2$$

On the other hand, under the condition given by Eq.(1.2), one element of z_j is given by the others: $(z_j)^{\kappa^\star} = 1 - \sum_{\kappa \neq \kappa^\star}^{\kappa} (z_j)^\kappa$ which implies that the theorical number of parameters for z_j is $\mathcal{N}_{\text{param},j}^{\text{theorical}} = (K-1) + \frac{M\cdot(M-1)}{2}(K-1)^2$ and by consequence:

$$\mathcal{N}_{\text{param}}^{\text{theorial}} = M(K-1) + \frac{M(M-1)}{2}(K-1)^2$$

2.2.5 Optimizer: Stochastic Gradient Descent (SGG)

Preliminary results [23] showed the efficiency of the stochastic gradient descent (SGD) as an optimizer for contact predictions and for the linear model case (in comparison to Adadelta, Adagrad and Adam). SGD is often used in machine learning to help with the minimisation of the loss function $\mathcal{L}$ measuring the difference between true and predicted value to extract the best parameters $\mathbf{W}$ such that $\mathcal{L}(\mathbf{W}) \overset{\text{epochs}\to\infty}{\longrightarrow} 0$. This optimizer used gradient descent method that consists to find "the steepest steps" between the input and the output [33]. However, the correct use of this optimizer requires to understand how the different step are computed regarding three factors : the learning rate, the momentum, and the nesterov (see visualisation on Fig.2.6).

Learning rate Each new step t travels from the previous one $t-1$ an amount of learning rate l_r with a direction orthogonal to the level curve (this means opposite to the gradient).

$$\Delta\mathbf{W}_{\text{step } t} = -l_r \frac{\partial\mathcal{L}(\mathbf{W}_{\text{step } t})}{(\partial\mathbf{W}_{\text{step } t})} \tag{2.11}$$

The choice of l_r for this optimizer is essential for good results to avoid the fall in a local mimimum or to not find a minimum which can happen if respectively l_r is to small or to high. Find the appropriate l_r means find a good balance between the simulation time and its accuracy. Preliminary results [23] showed that a learning

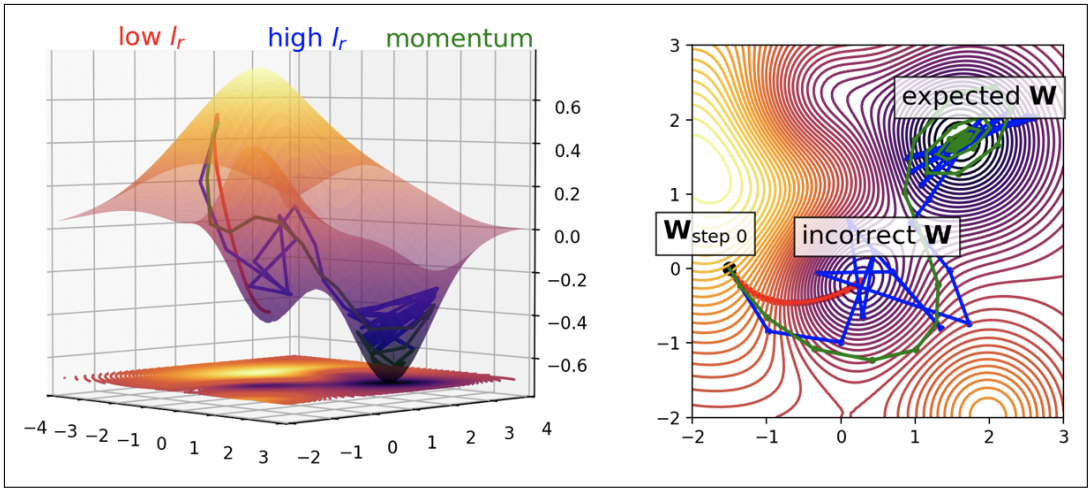


FIGURE 2.6: Different path according to the learning rate and momentum. A too low learning rate is not able to find the global minima. [7]

rate $l_r = 0.008$ was able to tend to the minimum without beeing blocked into local minima.

Momentum The learning can be speed-up with help of a momentum μ that will increase the learning rate for dimension with same direction of gradient and decrease it for dimension with several directions of gradient [33].

$$\Delta \mathbf{W}_{\text{step } t} = \mu \Delta \mathbf{W}_{\text{step } t-1} - l_r \frac{\partial \mathcal{L}(\mathbf{W}_{\text{step } t})}{(\partial \mathbf{W}_{\text{step } t})} \tag{2.12}$$

This equation showed that if the last direction $\Delta \mathbf{W}_{\text{step } t-1}$ is of same sign than $-l_r \frac{\partial \mathcal{L}}{(\partial \mathbf{W}_{\text{step } t})}$ the learning will be added to a fraction μ of the previous amount step. In contrast if they are not the same, the learning will be subtracted by a faction μ of the previous amount step. Comparison between different momentum values are generally made around $0.85 - 0.95$. The preliminary optimisation [23] didn't analyse SDG with momentum.

Nesterov The SGD optimization with nesterov doesn't look the direction of the gradient of the current step but of the current step with the momentum term:

$$\Delta \mathbf{W}_{\text{step } t} = \mu \Delta \mathbf{W}_{\text{step } t-1} - l_r \frac{\partial \mathcal{L}(\mathbf{W}_{\text{step }} + \mu \Delta \mathbf{W}_{\text{step } t-1})}{(\partial \mathbf{W}_{\text{step } t})} \tag{2.13}$$

The nesterov utilization will correct the new amount of step by looking the previous step (with the momentum term) and the future potential step [33]. The preliminary optimisation [23] used as default the nesterov .

2.3 Couplings and contacts

2.3.1 Contacts extraction and corrections

Let consider two amino acids $i \in [1, M]$ and $j \in [1, M]$ such that $i \neq j$. Their couplings strength are extracted from the weights $W_{ji}^{(1)}$ learned by the different models. Additionally, the over-parameterization is compensated with Ising gauge $\mathbf{W}_{ji}^{(1)} \overset{\text{Ising}}{\rightarrow} \mathbf{W}_{j,i}^{(1')}$ that enforces the average on the different values $\kappa \in [1, K]$ of j and on the different values $\phi \in [1, K]$ i to be zero. Additionally, the symmetry of the matrix is constrained by the complementary of the matrix transpose.

$$\mathbf{W}_{j,i}^{(1'')} = \frac{1}{2}\left(\mathbf{W}_{j,i}^{(1')} + (\mathbf{W}_{j,i}^{(1')})^{\mathrm{T}}\right) \qquad \text{symmetric} \tag{2.14}$$

Then the contacts between an amino acid i and j is computed with the Frobenius norm [34].

$$F_{j,i} = \sqrt{\sum_{\kappa,\phi}^{K} W_{j,i}^{(1'')}(\kappa,\phi)^2} \qquad \text{symmetric} \tag{2.15}$$

A final correction is done with an average correction that penalize correlations coming from phylogeny [35].

$$C_{j,i} = F_{j,i} - \frac{\sum_s F_{j,s} \sum_r F_{r,i}}{\sum_{r,s} F_{r,s}} \overset{F_{j,i}=F_{i,j}}{=\!=} F_{j,i} - \frac{(\sum_s F_{j,s})^2}{\sum_{r,s} F_{r,s}} \qquad \text{symmetric} \tag{2.16}$$

Ising and linear model It is possible to verify that with the following equation,the average over the different values of i ($*^6$) or of j ($*^7$) is null.

$$W_{ji}^{(1')}(\kappa^\star,\phi^\star) = W_{ji}^{(1)}(\kappa^\star,\phi^\star) - \frac{1}{K_j}\sum_{\kappa}^{K_j} W_{ji}^{(1)}(\kappa,\phi^\star) - \frac{1}{K_i}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa^\star,\phi) + \frac{1}{K_j K_i}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa,\phi) \tag{2.17}$$

Ising and linear cross proteins model This follows the same equation than Eq.(2.17) except for $(j,i) \in ([0, L_A] \times [0, L_A]) \cup ([L_A, L] \times [L_A, L])$ that are fixed to zero.

Ising and non linear model For the non linear model, the weights $\mathbf{W}_{ji}^{(1)}$ is impacted by the ising gauge applied on the second weights ($\mathbf{W}_{\mathbf{jiy}}^{(\mathbf{2})} \overset{\text{Ising}}{\rightarrow} \mathbf{W}_{\mathbf{jiy}}^{(\mathbf{2'})}$). It is possible to verify that with the following equation, the average over the different values of i ($*^8$), of j ($*^9$), or of y ($*^{10}$) is null.

The computations details $*^6$, $*^7$, $*^8$, $*^9$ and $*^{10}$ are found in Appendix.B

$$W_{jiy}^{(2')}(\kappa^\star,\phi^\star,\gamma^\star) = W_{jiy}^{(2)}(\kappa^\star,\phi^\star,\gamma^\star) - \frac{1}{K_j}\sum_{\kappa}^{K_j} W_{jiy}^{(2)}(\kappa,\phi^\star,\gamma^\star) - \frac{1}{K_i}\sum_{\phi}^{K_i} W_{jiy}^{(2)}(\kappa^\star,\phi,\gamma^\star)$$
$$- \frac{1}{K_y}\sum_{\gamma}^{K_y} W_{jiy}^{(2)}(\kappa^\star,\phi^\star,\gamma) + \frac{1}{K_jK_i}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W_{jiy}^{(2)}(\kappa,\phi,\gamma^\star) + \frac{1}{K_iK_y}\sum_{\phi}^{K_i}\sum_{\gamma}^{K_y} W_{jiy}^{(2)}(\kappa^\star,\phi,\gamma)$$
$$+ \frac{1}{K_yK_j}\sum_{\gamma}^{K_y}\sum_{\kappa}^{K_j} W_{jiy}^{(2)}(\kappa,\phi^\star,\gamma) - \frac{1}{K_jK_iK_y}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i}\sum_{\gamma}^{K_y} W_{jiy}^{(2)}(\kappa,\phi,\gamma)$$

The Ising gauge on $\mathbf{W}_{\mathbf{jiy}}^{(\mathbf{2})} \in \mathcal{R}^{K\times K\times K}$ implies the subtractions of three elements involving weights matrix of two dimensions.

$$\mathbf{W}_{jiy}^{(2,\kappa)} = \sum_{\kappa}^{K_j} W_{jiy}^{(2)}(\kappa,\phi^\star,\gamma^\star) \quad \forall\phi^\star,\gamma^\star \in [1,K] \quad \rightarrow\in \mathcal{R}^{KxK}$$
$$\mathbf{W}_{jiy}^{(2,\phi)} = \sum_{\phi}^{K_i} W_{jiy}^{(2)}(\kappa^\star,\phi,\gamma^\star) \quad \forall\kappa^\star,\gamma^\star \in [1,K] \quad \rightarrow\in \mathcal{R}^{KxK}$$
$$\mathbf{W}_{jiy}^{(2,\gamma)} = \sum_{\gamma}^{K_y} W_{jiy}^{(2)}(\kappa^\star,\phi^\star,\gamma) \quad \forall\kappa^\star,\phi^\star \in [1,K] \quad \rightarrow\in \mathcal{R}^{KxK}$$

By consequence, these ones need to be added to the 2D weights $\tilde{\mathbf{W}}_{ji}^{(1)}$ extracted in Eq.(2.5) in order to compensate their deduction:

$$\mathbf{W}_{ji}^{(1)} = \tilde{\mathbf{W}}_{ji}^{(1)} + \mathbf{W}_{jiy}^{(2,\kappa)} + \mathbf{W}_{jiy}^{(2,\phi)} + \mathbf{W}_{jiy}^{(2,\gamma)}$$

Then the ising gauge can be applied as in Eq.(2.17).

2.4 Errors propagation

The errors propagation transmitted to the final couplings can be computed considering the absolute error of a function $\mathcal{F}(\boldsymbol{x})$ with P parameters x_p as :

$$\Delta\mathcal{F}(\boldsymbol{x}) = \sum_{p}^{P} |\frac{\partial\mathcal{F}}{\partial x_k}| \cdot \Delta x_p \tag{2.18}$$

2.4.1 Absolute errors

Absolute error of the weights $W_{ji}^{(1)}$ before ising Let's consider the K elements $W_{ji^\star}^{(1)}(\kappa,\phi^\star)$ given in Eq.(2.5), their absolute errors are given by:

$$\begin{cases} \Delta W_{ji^\star}^{(1)}(\kappa,\phi^\star) \overset{(2.18)}{=} \Delta(z_j(\kappa))^{(l)} + \Delta W_j^{(0)} & \forall\kappa \in [1,K], \text{and } j \in [1,M] \\ (a_i^\phi)^l = \delta(i,i^\star)\cdot\delta(\phi,\phi^\star) & \forall\phi \in [1,K], \text{and } i \in [1,M] \end{cases}$$

with the K absolute error of the biases $W_j^{(0)}(\kappa)$ given in Eq.(2.6):

$$\begin{cases} \Delta W_j^{(0)}(\kappa) = \Delta(z_j(\kappa))^l & \forall \kappa \in [1,K], \text{and } j \in [1,M] \\ (a_i^{\phi})^l = 0 & \forall \phi \in [1,K], \text{and } i \in [1,M] \end{cases}$$

The K absolute errors of the probabilities $(z_j(\kappa))^l$ can be computed thanks to their relation with the K predictions y_j^{β} given in Eq.1.11:

$$\begin{cases} (z_j^{\beta})^l = \log(\hat{y}_j^{\beta})^l (\sum_{\beta}^{K} \exp(z_j^{\beta})^l) = \log((\hat{y}_j^{\beta})^l) + (C_j^{\beta})^l & \text{if } (\hat{y}_j^{\beta})^l \neq 0 \\ (z_j^{\beta})^l = 0 & \text{if } (\hat{y}_j^{\beta})^l = 0 \end{cases}$$

$$\begin{cases} \Delta(z_j^{\beta})^l \overset{(2.18)}{=} \left| \frac{1}{(\hat{y}_j^{\beta})^l} \right| \Delta(\hat{y}_j^{\beta})^l & \text{if } (\hat{y}_j^{\beta})^l \neq 0 \\ \Delta(z_j^{\beta})^l = 0 & \text{if } (\hat{y}_j^{\beta})^l = 0 \end{cases}$$

Absolute error of the weights $W_{ji}^{(1')}$ after ising The $K \times K$ absolute error on the weights after Ising $W_{ji}^{(1')}(\kappa^\star, \phi^\star)$ can be computed with an adaptation of Eq.(2.17) $(*^{11})$: :

$$\begin{aligned} \Delta W_{ji}^{(1')}(\kappa^\star,\phi^\star) &\overset{(2.18)}{=} \left| \frac{\partial W_{ji}^{(1')}(\kappa^\star,\phi^\star)}{\partial W_{ji}^{(1)}(\kappa^\star,\phi^\star)} \right| \Delta W_{ji}^{(1)}(\kappa^\star,\phi^\star) + \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} \left(\left| \frac{\partial W_{ji}^{(1')}(\kappa^\star,\phi^\star)}{\partial W_{ji}^{(1)}(\kappa,\phi^\star)} \right| \Delta W_{ji}^{(1)}(\kappa,\phi^\star) \right) \\ &+ \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_i} \left(\left| \frac{\partial W_{ji}^{(1')}(\kappa^\star,\phi^\star)}{\partial W_{ji}^{(1)}(\kappa^\star,\phi)} \right| \Delta W_{ji}^{(1)}(\kappa^\star,\phi) \right) + \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_i} \left(\left| \frac{W_{ji}^{(1')}(\kappa^\star,\phi^\star)}{\partial W_{ji}^{(1)}(\kappa,\phi)} \right| \Delta W_{ji}^{(1)}(\kappa,\phi) \right) \\ &= \left| 1 + \frac{1}{K_j K_i} - \frac{1}{K_j} - \frac{1}{K_i} \right| \Delta W_{ji}^{(1)}(\kappa^\star,\phi^\star) + \left| -\frac{1}{K_j} + \frac{1}{K_j K_i} \right| \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} \Delta W_{ji}^{(1)}(\kappa,\phi^\star) \\ &+ \left| -\frac{1}{K_i} + \frac{1}{K_j K_i} \right| \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_j} \Delta W_{ji}^{(1)}(\kappa^\star,\phi) + \left| \frac{1}{K_j K_i} \right| \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_i} \Delta W_{ji}^{(1)}(\kappa,\phi) \end{aligned}$$

Absolute error of the symmetric weights $W_{j,i}^{(1'')}$ The absolute error of the symmetric weights $W_{j,i}^{(1'')}$ given in Eq.(2.14) is computed as:

$$\Delta W_{j,i}^{(1'')} = \frac{1}{2}(\Delta W_{j,i}^{(1')} + \Delta(W_{j,i}^{(1')})^{\mathrm{T}})$$

The computations details $*^{11}$ are found in Appendix.C

Absolute error of the Frobenius norm $F_{j,i}$ The absolute error of $F_{j,i}$ given in Eq.(2.15) is computed as:

$$\Delta F_{j,i} \overset{(2.18)}{=} \sum_{\lambda}^{K_j}\sum_{\mu}^{K_i}\left(\left|\frac{W_{j,i}^{(1'')}(\lambda,\mu)}{\sqrt{\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W_{j,i}^{(1'')}(\kappa,\phi)^2}}\right| \Delta W_{j,i}^{(1'')}(\lambda,\mu)\right)$$
$$= \frac{1}{F_{j,i}}\sum_{\lambda}^{K_j}\sum_{\mu}^{K_i}\left(\left|W_{j,i}^{(1'')}(\lambda,\mu)\right| \Delta W_{j,i}^{(1'')}(\lambda,\mu)\right)$$

Absolute error of couplings $C_{j,i}$ The couplings $C_{j,i}$ given in Eq.(2.16) can be reformulated as:

$$C_{ji} = F_{j,i} - \frac{(\sum_s F_{j,s})^2}{\sum_{r,s} F_{r,s}} = F_{ji} - \frac{(F_{ji} + \sum_{s\neq i}^{M} F_{js})^2}{F_{ji} + 2\sum_{s\neq i}^{M} F_{js} + \sum_{r\neq j}^{M}\sum_{s\neq i}^{M} F_{rs}}$$

which gives the absolute error :

$$\Delta C_{ji} \overset{(2.18)}{=} \left|\frac{\partial C_{j,i}}{\partial F_{ji}}\right| \Delta F_{ji} + \sum_{s\neq i}^{M}\left(\left|\frac{\partial C_{ji}}{\partial F_{js}}\right| \Delta F_{js}\right) + \sum_{r\neq j\neq i}^{M}\sum_{s}^{M}\left(\left|\frac{\partial C_{ji}}{\partial F_{rs}}\right| \Delta F_{rs}\right)$$

with $(*^{12})$:

$$\begin{cases} \dfrac{\partial C_{j,i}}{\partial F_{ji}} = 1 - \dfrac{2\sum_s F_{j,s}(\sum_{r,s} F_{r,s} - \sum_s F_{j,s})}{(\sum_{r,s} F_{r,s})^2} & \\ \dfrac{\partial C_{ji}}{\partial F_{js}} = \dfrac{2\sum_s F_{j,s}(\sum_{r,s} F_{r,s} - 2\sum_s F_{j,s})}{(\sum_{r,s} F_{r,s})^2} & \forall s \in [1,M]/i \\ \dfrac{\partial C_{ji}}{\partial F_{rs}} = \dfrac{(\sum_s F_{j,s})^2}{(\sum_{r,s} F_{r,s})^2} & \forall r \in [1,M]/j \end{cases}$$

2.4.2 Gaussian error with neighboring contacts

Since the method behind this algorithm is based on homologous sequences, some regions can be highly random and others very similar (see Fig.2.7).

This in-homogeneity can reveal noisy areas on the final map contact plot and induce wrong contact predictions. These areas can be revealed by considering not only the error of a position but also the noise of the neighbors.

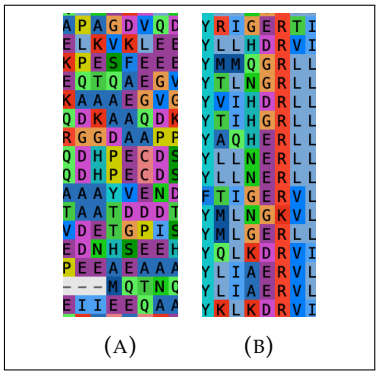


FIGURE 2.7: Different positions alignment in the homologous sequences of GrpE with AliView [25].

The computations details $*^{12}$ are found in Appendix.C

This is done by drawing N_s squares around each positions (i, j) and by attributing a coefficient α_i for each square i. These coefficients are given according to a gaussian with maximum value 1 and a standard deviation σ chosen as function of the numbers of squares.

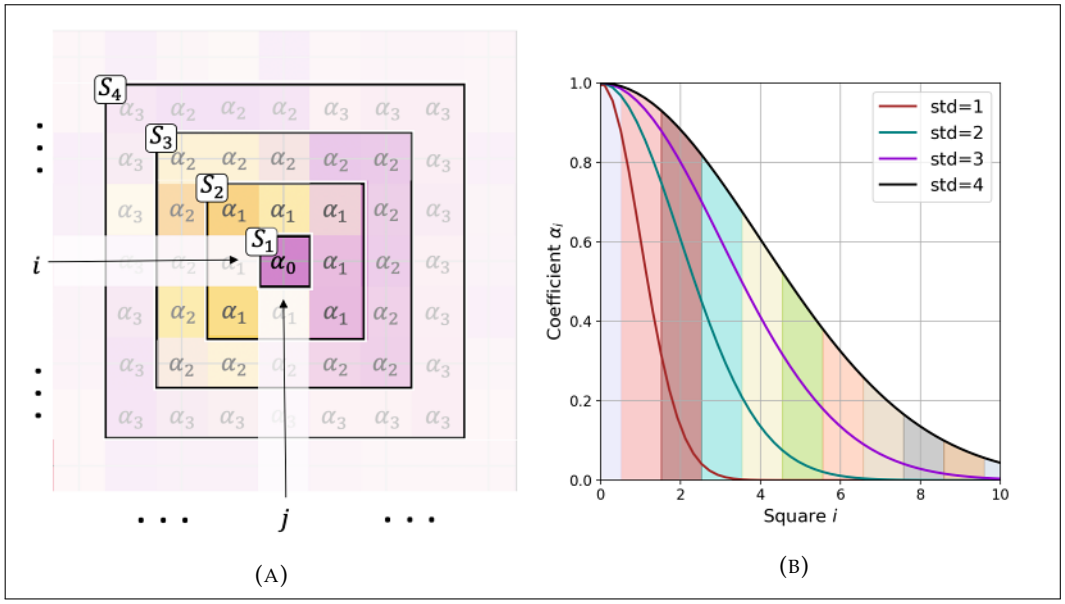


FIGURE 2.8: Representation of the errors neighbors influence with Eq.(2.24b).[7]

The new error $\mathcal{G}(i, j)$ for the pair and the coefficients are given by:

$$\begin{cases} \mathcal{G}(i,j) = \sum_{i}^{N_s} g_i \left(\sum_{\substack{(a,b) \\ \text{in square i}}} \Delta C_{ab} \right) & \text{(2.24a)} \\ \alpha_i = \exp\left(-\frac{i^2}{2 \cdot \sigma^2}\right) & \text{(2.24b)} \end{cases}$$

2.5 Map contact

Thanks to the next-generation molecular visualization program Chimerax [36], it is possible to visualise the 3D structure of a protein given by alphafold. This program allows several tools as the distance measurement between two aminos. However, the 2D map from alphafold has to be created and the contacts predictions have to be compared with this one.

These two process are done thanks to two python files (*mapPDB* and *PlotTopContacts*) belonging to a collection of tools writting by Duccio Malinverni [37]. However, note that these files have been modified to produce the results presented in the next chapter.

Modifications in mapPDB:

- The atoms of oxygen (O) and of carbon (C) are not anymore considered during the extraction of the minimal distance between each atoms of two proteins.

These ones seemed not consider during the minimal distance computed on Chimerax.

Changments in PlotTopContacts:

- Possibility to visualise the monomer and dimer contacts, or a superposition of two maps contacts between a protein and a dimer protein
- Possibility to have the errors map with Gaussian format or not, and to penalize the contacts predictions
- Possibility to choose only specific areas for the contacts predictions
- Adapted to the case of contacts predictions from a cross proteins model

2.5.1 Proteins pairing

Thanks to Uniprot [29], it is possible to extract several sequences of proteins. However, usually there are more than one sequence per organism which complicates the porteins pairing. Whereas, Uniprot [29] can sometimes mentioned which sequences are known to be in interaction, there exist no known database able to extract a quantity of sequences pairs between two types of proteins.
In bacteria, relative gene are often found in the same region of chromosome controlled by a common operon [38]. Hopefully, Uniprot gives acces to the Ordered Locus Names (OLNs) (and/or the open reading frames (ORFs)) that are identified by a code name containing the strand and the ordering of genes on the chromosome [**?**]. This gives the following computational pairing procedure:

- The reference pairs is given by the pairs of the two references:
  $S_{AB}^{\text{ref}} = S_A^{\text{ref}} : S_B^{\text{ref}}$
- Formation of groups of same organisms $\mathcal{O}$ between sequence $S_A^{\mathcal{O}}$ from protein A and sequence $S_B^{\mathcal{O}}$ from protein B.
- Computation of each possible pairs $S_A^{\mathcal{O}_\Omega} : S_B^{\mathcal{O}_\Omega}$ that have the same strand Ω defined in OLNs (or the same strand defined in ORFs). For $L_A^{\mathcal{O}_\Omega}$ and $L_B^{\mathcal{O}_\Omega}$ sequences there are $L_A^{\mathcal{O}_\Omega} \cdot L_B^{\mathcal{O}_\Omega}$ configurations possible. Don't add a pair if there exist another one similar at X%.
- Measurement of the absolute distance of each pairs by using the accessible OLNs numbers (or ORFS numbers)
- The pairs with a chromosomal distance smaller than *MinDistance* remain and the others are removed

In Eukaryota, there are not (or rarely) operons; the pairs are formed randomly (but of same organism).

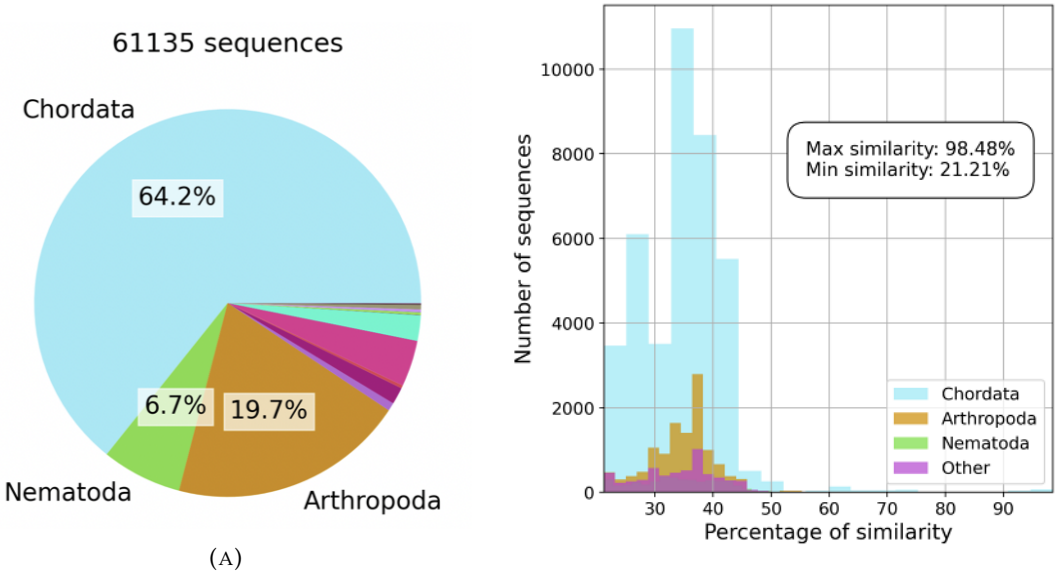
Chapter 3

The J domain (DnaJ) - IPR001623

the J domain (DnaJ) has a small number of amino acids in comparison to the other types of proteins treated in the next parts. Therefore the training of the model is less time and computational costly, making it a good sample to compare different experiments.

3.1 Processing of data

Only the DnaJ from metazoa have been extracted and analysed. The reference protein is the DnaJ homolog subfamily C member 30, mitochondrial (66 amino acids, Q96LL9· DJC30, organism Homo sapiens (Human) and phylum Chordata). Since the data contain an important number of sequences (71'797) and that some of them should maybe not be considered, a reduction to 61'135 sequences ($\sim$ 85%) is applied by fixing a minimum similarity with the first sequence to 20% of the same amino acids. The different similarities according to different phylum (with the reference sequence) are shown on Fig.3.1b and reveal a global average around 38%. The different phylum proportion inside these data are shown on Fig.3.1a.



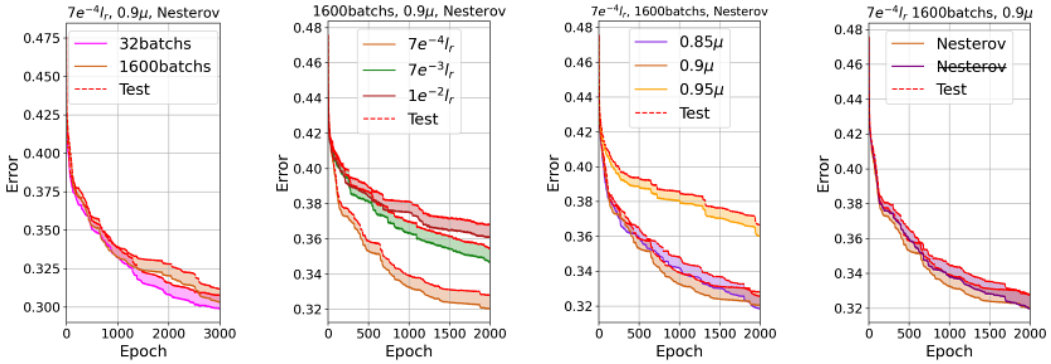
(A) (B)

FIGURE 3.1: (A) Phylum distribution and (B) distribution of similarity with the reference sequence Q96LL9· DJC30 (after a filtering of 20% min).

3.2 Learning influence from parameters

3.2.1 Different learning and testing curves

Four different factors have been studied and comparisons are displayed on Fig.3.2.



(A) (B) (C) (D)

FIGURE 3.2: Training and testing curves with different (A) learning rates, (B) batchs, (C) momentum, and (D) with or without Nesterov.

The different curves of learning rate l_r on Fig.3.2b highlight a smaller minimum with $7e^{-4}l_r$. On the other hand, increasing the number of batches (see Fig.3.2a) results in a increase of error in the beginning of the epochs before reaching similar errors at 3000epochs (1.25% differences at 2000 epochs between the test curves and 0.04% differences at 2000 epochs between the test curves). The different momentum μ on Fig.2.12 point out the importance to select the appropriate one. Indeed, a too big momentum (see orange curve) can decrease the performance of the model. Finally, the nesterov doesn't show large difference for $\mu = 0.9$.

3.2.2 Different contacts maps

The previous results led the next of the experiment by comparing the amino acids and contacts predictions (Fig.3.3) between two distinct learning rates, $1e^{-2}$ and $7e^{-4}$, to reveal the impact of a better learning curve. The following parameters are taken when training the model: 30′000 epochs, 1′600 batchs, 0.9μ, Nesterov, and the learning curves are found on Fig.3.3c. The reason behind the batch size choice is to save computational time since the distinction between 32 and 1600 batchs demonstrated no significant difference in the accuracy of the model. This choice of epochs is made to tend towards the end part of the decreasing learning curve of $0.0007l_r$. Observe that the curve of $0.01l_r$ already achieves its minimum at around 8′500 epochs. Additionally, the error difference between the test curves at 30′000 epochs is approximately 7.5%.

The contacts predictions on Fig.3.3a and 3.3b accentuates the importance in fine-tuning the parameters of the model before starting the experiments. Indeed, the 7% error difference between the test curves leads to 35% difference in correct contacts predictions. Furthermore, a final comparison is done by taking not only one model with $0.0007l_r$ (and the parameters mentioned above) but eight. These eights models

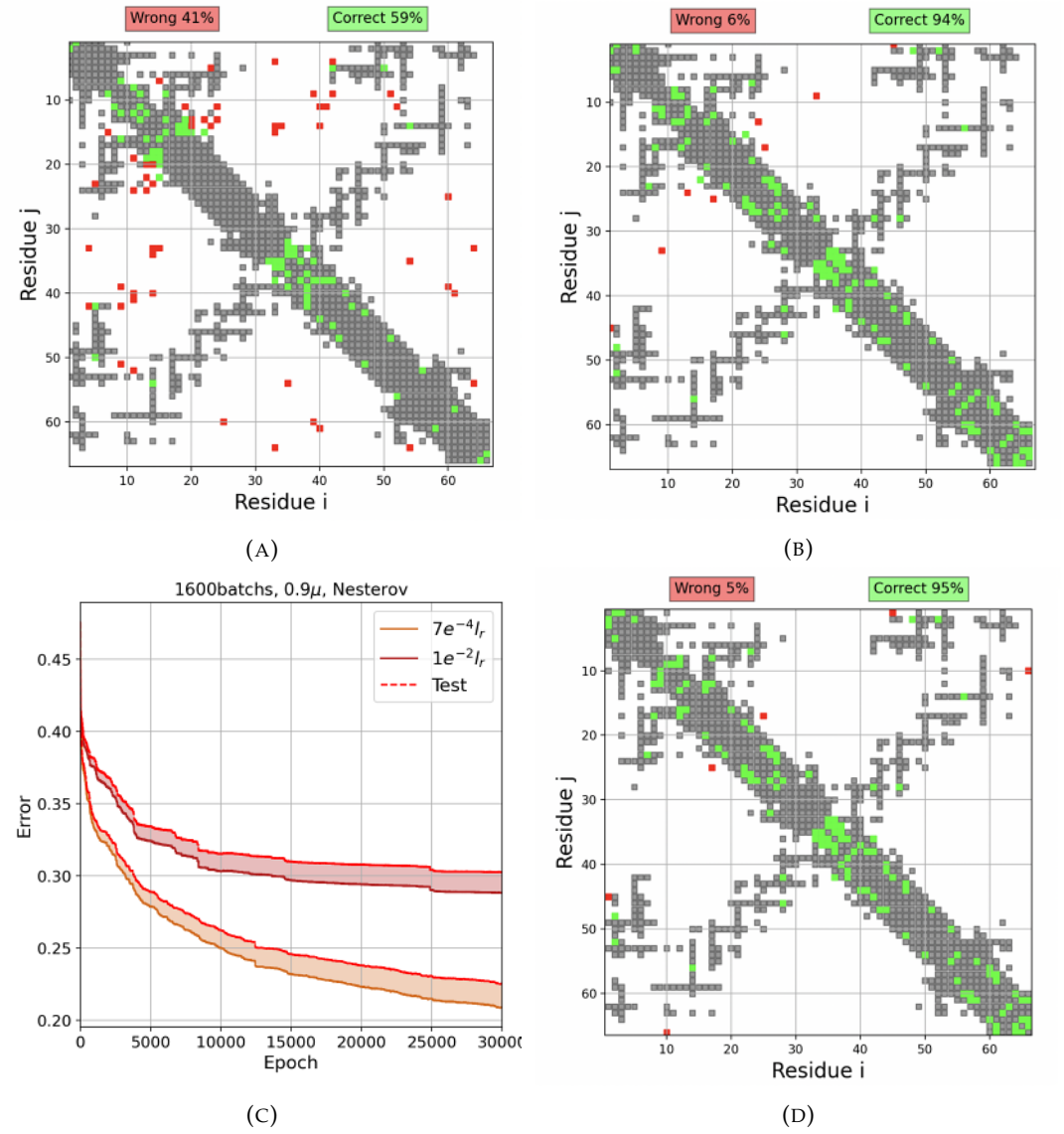
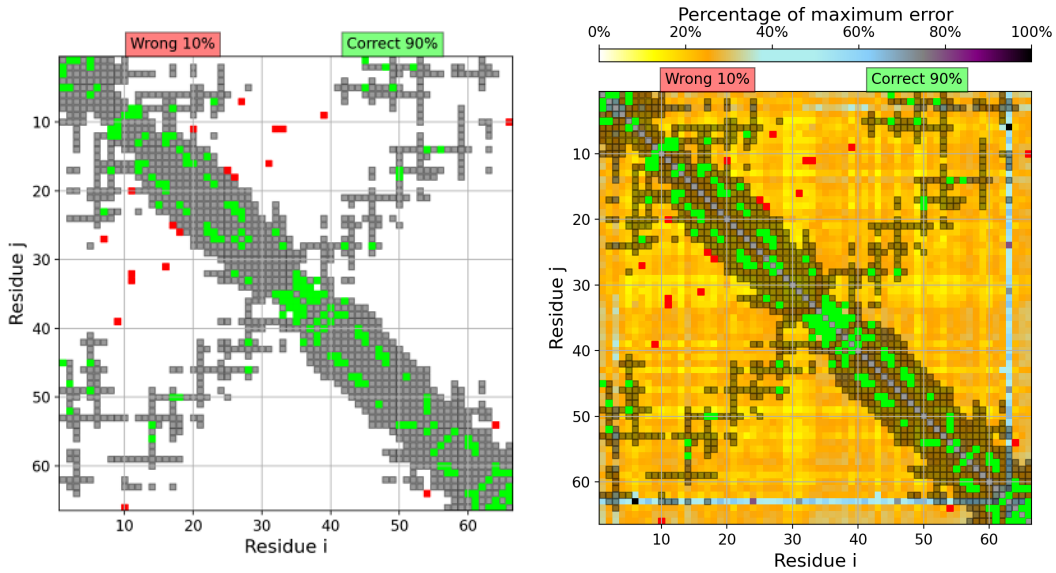


FIGURE 3.3: Difference in 80 contacts predictions between models with different minimisation (A/B), and effect from average over the 8 couplings from 8 models (C). The models parameters are: $0.01l_r$, 1600batchs, 0.9μ, Nesterov (A) and $0.0007l_r$, 1600batchs, 0.9μ, Nesterov (B,C). The threshold of contact is fixed to 8.5Å.

start with different orders and distributions of training (and test) data. The difference in contact predictions between Fig.3.3b and 3.3d reveals an improvement in 94% to 95% correct contacts predictions.

3.3 Error correction

Although the previous experiment shows that it is possible to reach 95% of correct contacts predictions, it is not anymore the case with higher numbers of predictions. Increase the numbers of predictions leads in an invasion of new noisy contacts. A first comparison shows no improvement between the map with original contacts scores and the map with scores subtracted with their absolute errors (without gaussian error consideration) (see Fig.3.4).



(A) (B)

FIGURE 3.4: (A) Contact map with errors of 100 predictions with a threshold fixed to 8.5Å. (B) Resulting contact map by decreasing the predictions with their errors.

By comparing the contact that are not on the diagonal between Fig.3.4 and Fig.3.3d, five new good predictions are observed on the bottom left-hand corner. The 15 others new predictions are on the diagonal or are possibly noise. Note the region between the amino acids $a_{27}, .., a_{35}$ and $a_{10}, .., a_{20}$ that regroups a lot of wrong predictions.

3.3.1 Gaussian errors

A second comparison is done by using Gaussian errors and four different pairs of (N_s, σ) on Fig.3.6. Please refer to Eq.(2.24b) for the definition of these terms. The different coefficient α_i corresponding to these specific pairs are shown on Fig.3.5.

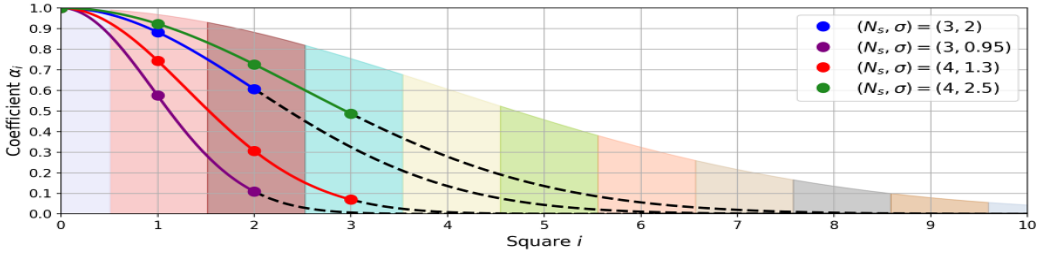


FIGURE 3.5: Five different pairs (N_s, σ) and their corresponding coefficient α_i for the Gaussian errors in Eq.(2.24b)

It is observed that for a higher number of squares, the error map is smoother. Note also the lost of information (including two good predictions in the left-hand corner and others in the diagonal corners) in the border for $N_s \leq 2$.

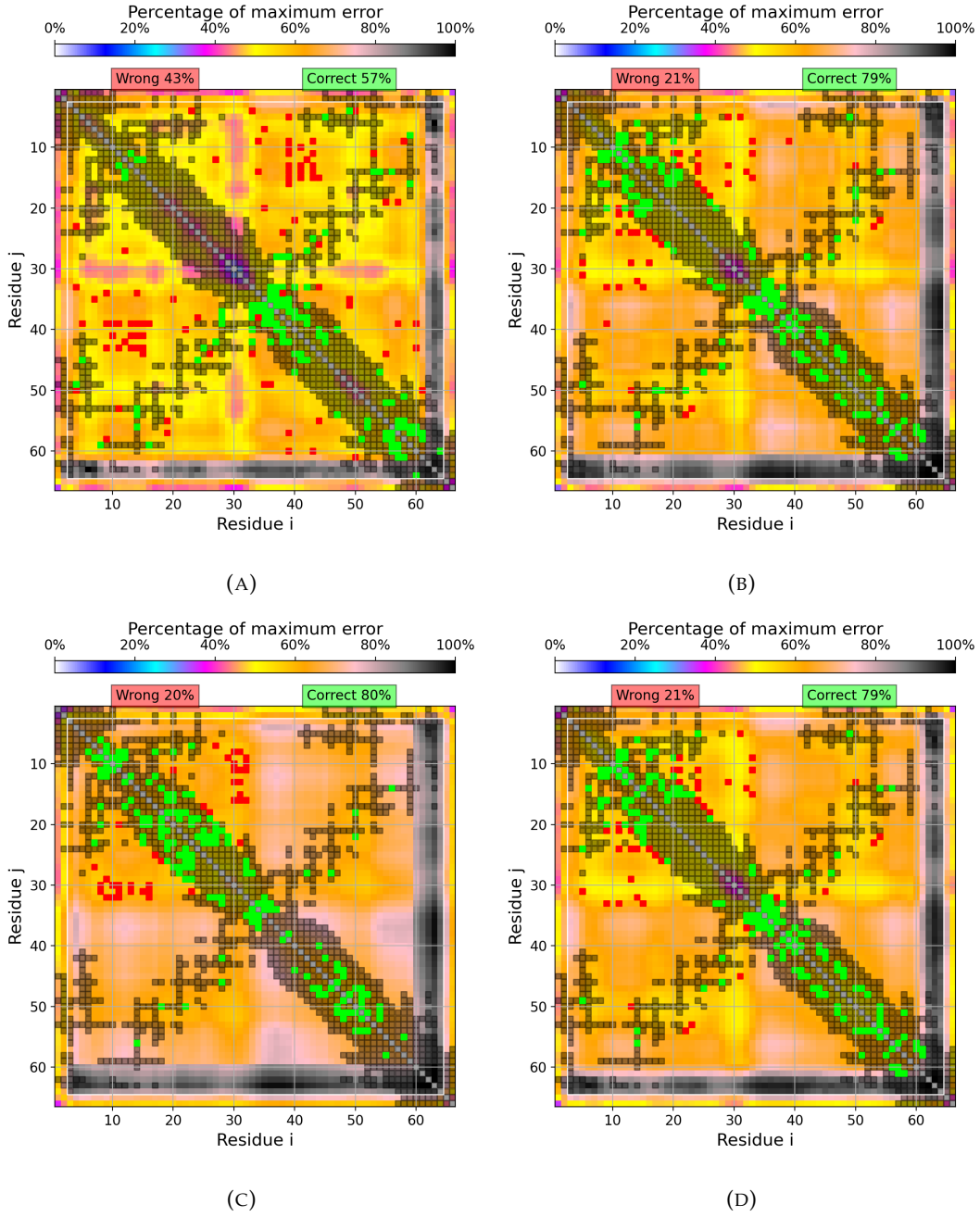


FIGURE 3.6: 100 contacts predictions and threshold 8Å with Gaussian error consideration and (N_s, σ) given by A:$(3, 0.95)$, B:$(3, 2)$, C:$(4, 2.5)$, and D:$(4, 1.3)$. The white rectangles visualise which contacts can be possible (no contact in the border of $\pm\lfloor N_s/2 \rfloor$).

The first case, (N_s, σ)=$(3, 0.95)$, points out that the results can be worse by giving high importance to the neighbors in the first square S_1 and very low (or none) interest to the next ones. Although, the noise between the amino acids $a_{27}, .., a_{35}$ and $a_{10}, .., a_{20}$ seems to be fixed, other noises appear in other regions. Furthermore a high number of correct predictions in the diagonal part, and also between the amino acids $a_{15}, .., a_{25}$ and $a_3, .., a_{10}$ disappear. The opposite situation is found with (N_s, σ)=$(4, 2.5)$

by considering a high percentage errors from the three nearest squares S_1, S_2, S_3. Indeed, the noise between the amino acids $a_{27}, .., a_{35}$ and $a_{10}, .., a_{20}$ is not fixed but accentuated. The remaining pairs (N_s, σ)=(3, 2) and (N_s, σ)=(4, 1.3) show similar results but have still a worst accuracy than the one on Fig.3.4b. However, note that new contacts are present at the bottom left-hand corner.

3.3.2 Selective regions

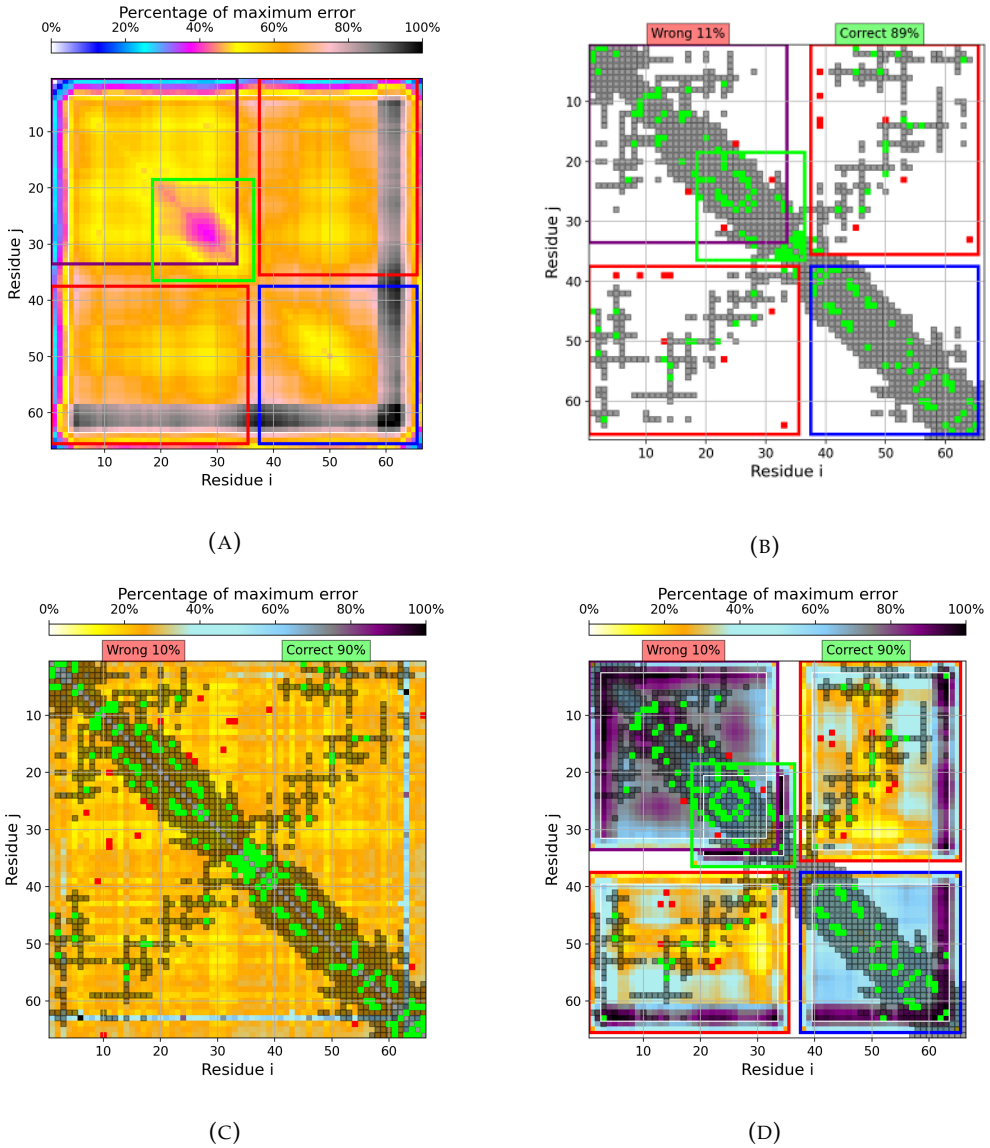


FIGURE 3.7: (A) Gaussian errors noise with $N = 5$ and $\sigma = 6$ and delimitation of region presented particular variation. Resulting contacts predictions (B) without errors consideration or (D) with Gaussian errors consideration. (C) 100 contacts predictions.

The final experiment of this subsection consists in determining if the noise reduction can be improved by considering only specific regions of the plots. Two errors maps are presented on Fig.3.7 and the variation areas are underlined with different colored

rectangles. The numbers of predictions between the squares are not equal because for N predictions, the symmetric region in the diagonal part actually gives only $N/2$ contacts (the other half are for the same pairs). Thus 50 contacts dots were predicted in the symmetric squares and 25 in the two external red rectangles. Note also that the total number of predictions is not strictly identical to the sum of the predictions asked in each regions since some predictions can be the same between two squares overlapping. The percentages of errors without regions selected (Fig.3.7b) or with selection and Gaussian errors (Fig.3.7d) are equal. But observe how the numbers of correct predictions at the bottom left-hand corner increased by using regions represented on Fig.3.7d and 3.7c.

3.4 Model's accuracy

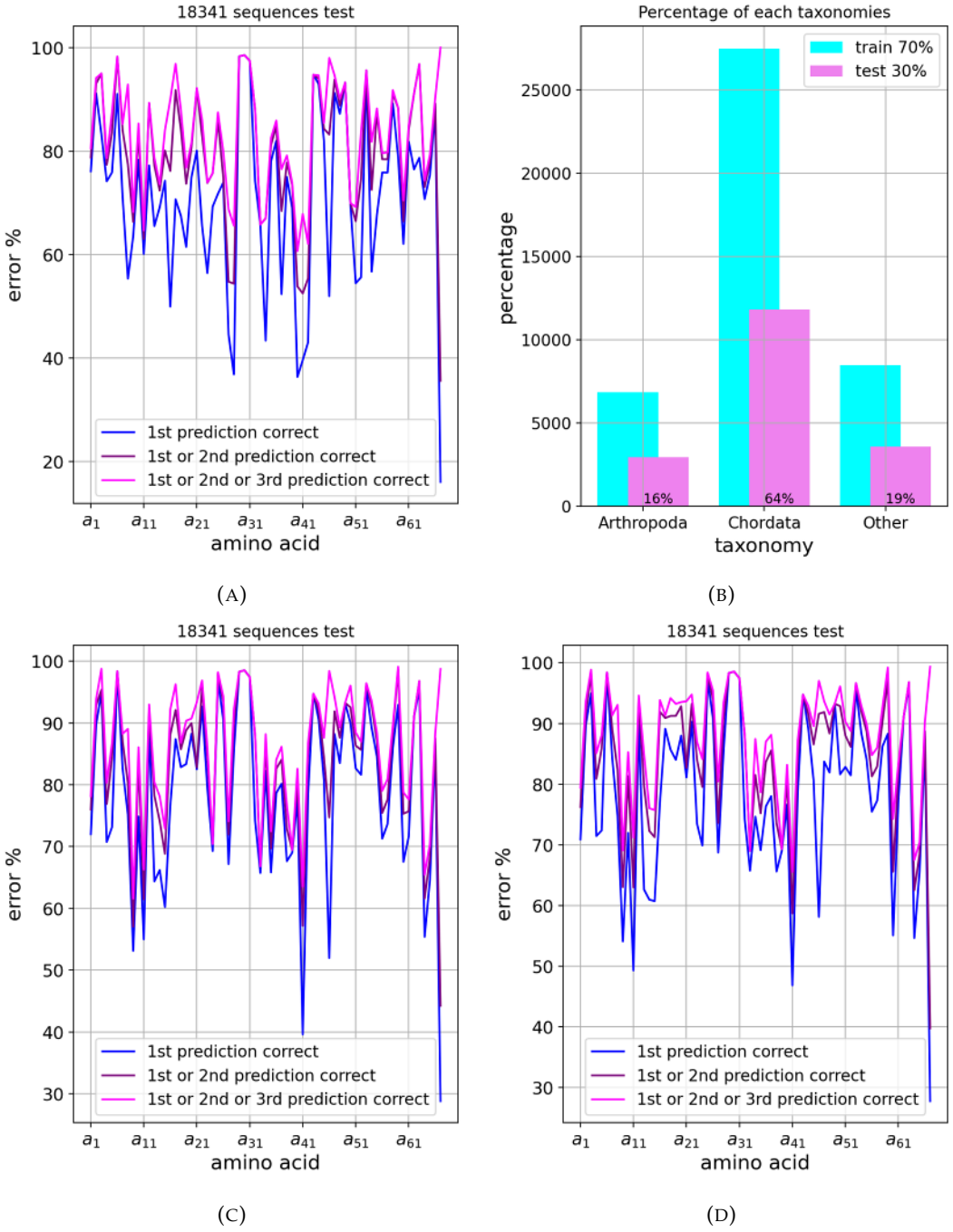
The previous models that has led to the couplings and the contacts maps can also be exploited to predict one amino acid with the others from the sequence.

3.4.1 Errors percentage

The percentage of correct predictions are done over the test data and positions including the taxonomy (corresponding to the last position) (see Fig.3.8). The improvement between $0.01 l_r$ to $0.0007 l_r$ is visible with a general accuracy increment. However note how the positions 11 and 41 are affected on Fig 3.8c. These errors could explain wrong contacts on Fig.3.4b aligned with these coordinates. Whereas the first prediction doesn't appear to be highly improved by taking an average over 8 models, the second predictions show better results (Fig.3.8d). Finally, the accuracy of the taxonomy predictions is poor in every cases.

3.4.2 Amino acids predictions

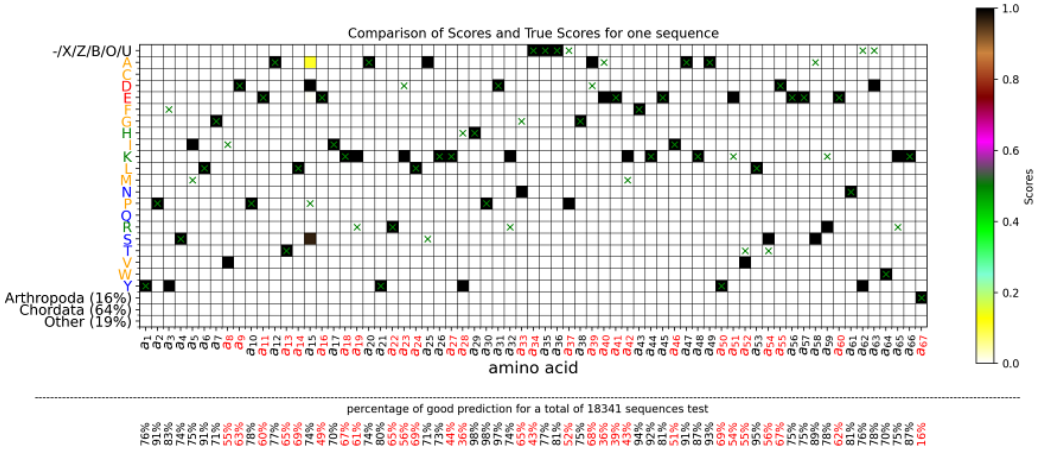
The predictions maps, illustrated on Fig.3.9, highlights the worst accuracy for $0.01 l_r$ (see the percentage errors numbers on the bottom) and the difficulty to find the correct taxonomy. The difference of probability between the three models can be visualise with the color map for one sequence. The position that don't show the value for the higher probability are a_4, a_5, a_{11}, a_{13}, a_{15}, a_{23}, a_{25}, a_{28}, a_{37}, a_{40}, a_{51}, a_{52}, a_{54}, a_{58}, a_{59}, a_{62} and a_{64}. The analysis on the predictions for the negative amino acids (aspartic acid D or glutamic acid E) reveals some difficulty of the model to choose between them or others values as the potassium (K), the serine (S) and the threonine (T). The positions a_4 presents a small likelihood ($\leq 40\%$) to be a D or E with the best models (Fig.3.9b & 3.9c). But the higher probability shows in this position with the bests models is a potassium (K). However the true value should be a serine (S); that was correctly predicted with the worst model on Fig.3.9a. Observe also the position a_{11} that was correctly predicted to be E on Fig.3.9a but that was showed to have the best probability to be D and smallest probability ($\leq 20\%$) to be S or T on Fig3.9c. The position a_{23} should be D but has a small probability ($\sim 20\%$) and the best probability to be E on respectively Fig.3.9c and Fig.3.9b. This position has also the best probability to be K on Fig.3.9a and Fig.3.9c.



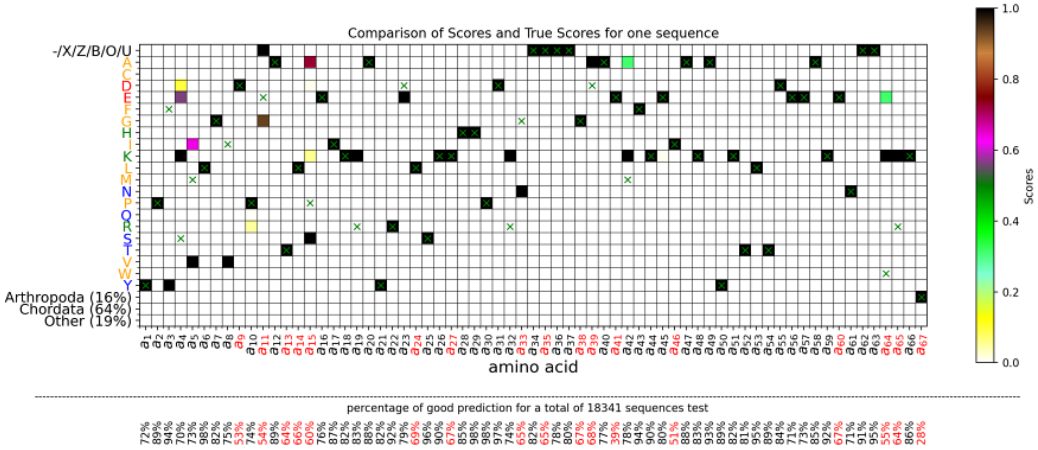
(A) (B)

(C) (D)

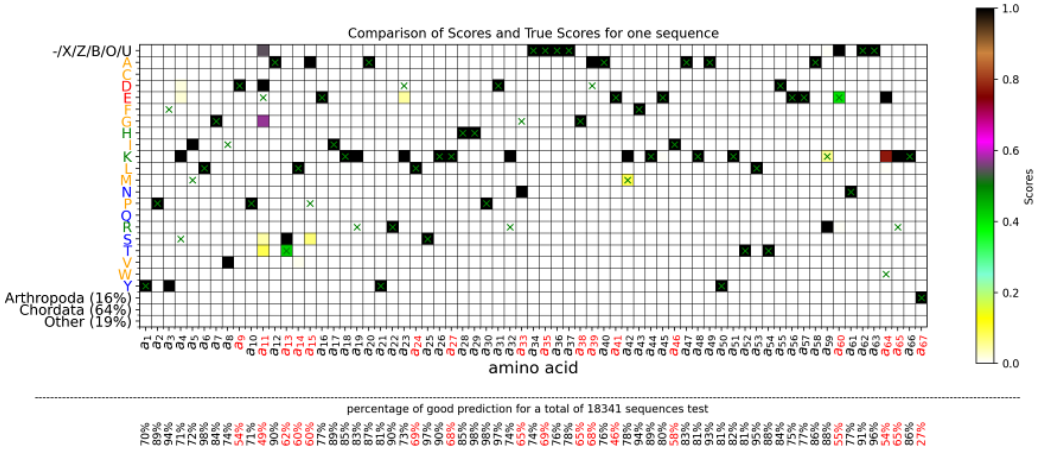
FIGURE 3.8: (A/C) $l_r = 7e^{-4}$ 1model/ average over 8models, and (D) $l_r = 1e^{-2}$. Distribution of the taxonomy in (B).



(A)



(B)



(C)

FIGURE 3.9: (A/C) $l_r = 0.01$ 1model/ average over 8models, and (B) $l_r = 0.0007$. The sum of the K predictions is 1 in every position. The green cross is the true value. polar:yellow, apolar:orange, negative:red, positive:green. The percentage in red is $\leq 70\%$

3.5 Discussion

The higher minimum reached by the curves with $l_r = 7e^{-3}$ and $l_r = 1e^{-2}$ (Fig.3.2b) could be the cause of too high value making the convergence towards the global minimum impossible as illustrated Fig.2.6. The wrong contacts found with $l_r = 7e^{-4}$ are very likely caused by this too high learning rate that has found a local minimum. The model learning was blocked in a region where only $\sim$ 70% of the amino acids could be predicted by the others (Fig.3.3c), and causing at least 30% of wrong couplings C_{ij}(i<j) predictions that appear as noise on the contact map (Fig.3.3a). The wrong contacts are possibly attenuated with the average on different couplings thanks to noise compensation. Since they don't learn on the same distribution and order of sequences, the noise accumulated during the learning of one model will be attenuated with the other models .

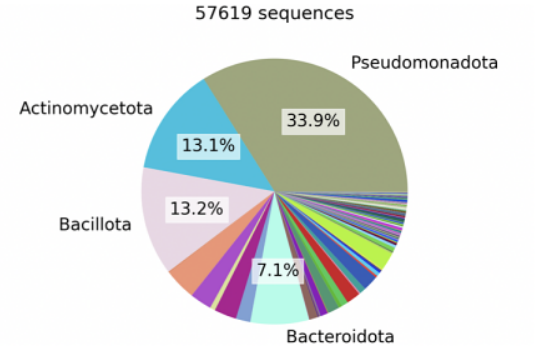
The consideration of the absolute errors for the whole map with or without Gaussian (Fig.3.4b and Fig.3.6) does not seem to increase the accuracy of contacts predictions. However the strategy to use the Gaussian errors to have a map with the error variation seems to be helping in determining which regions require an analysis. Additionally, the results of restrained regions demonstrates an improvement in contacts predictions for region outside of the diagonal (Fig.3.7d). This new contacts were likely hidden by the strong contacts in the diagonal part or in the noisy regions.

The Chordata dominance in the data distribution (Fig.3.8b) could be the reason of the poor taxonomy accuracy (27%) observable over the 18'341 sequences that were not used during the model training (Fig.3.9). In order to avoid the training over only one type of taxonomy, the distribution of sequences for the learning phase was done with the same percentage of each group of phylum type (here $\sim$ 33%) but this conducted to a majority of Chordata sequences since this kind was more than 64% of the dataset (Fig.3.1). A modification of the data division could be to take the same amount of sequences per phylum type. However this could rise on a colossal diminution of the training sequences that could not be enough for a high accuracy. Another idea would be to assign different weights for the taxonomy type as it has been done for sequences that are too similar in the dataset.

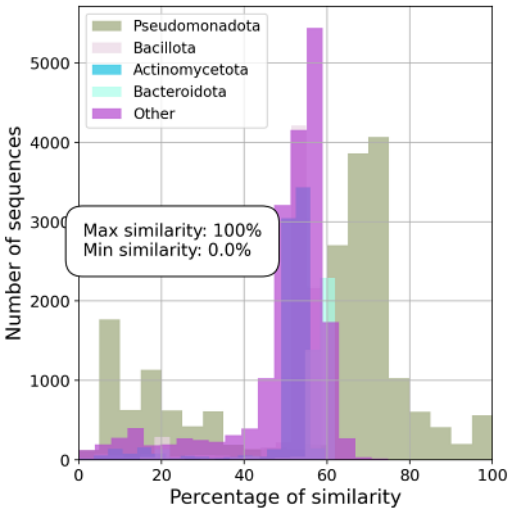
Although the taxonomy prediction was not optimal, the amino acids predictions highlight a high numbers of position with accuracy larger than 70% (50/67 for the model(s) with $7e^{-4}l_r$ on Fig.3.9b & 3.9c). The connection between the negative amino acids probabilities and other types (polar: K, no-polar S,T) could come from a coincidence or could point out a transformation between homologous sequences through evolution. Such transformation could be for example phosphorylation that involves in majority the Serine S and then the Threonine T (and the Tyrosine Y) [39]. However nothing can be assert with the observation done only on one sequence and external work could investigate this aptitude through the all dataset.

Chapter 4

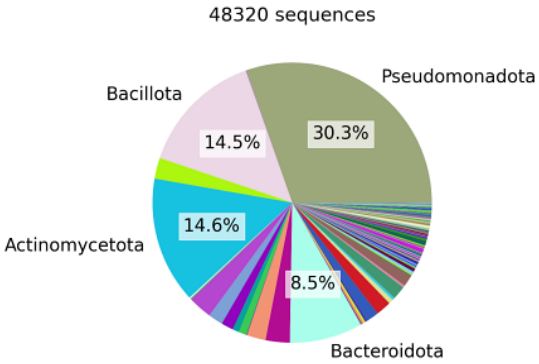
Gro-P like E (GrpE) and prokaryotic Hps70 (DnaK)



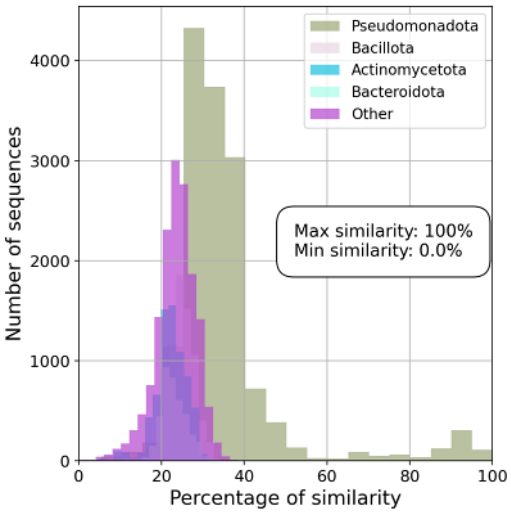
(A)



(B)



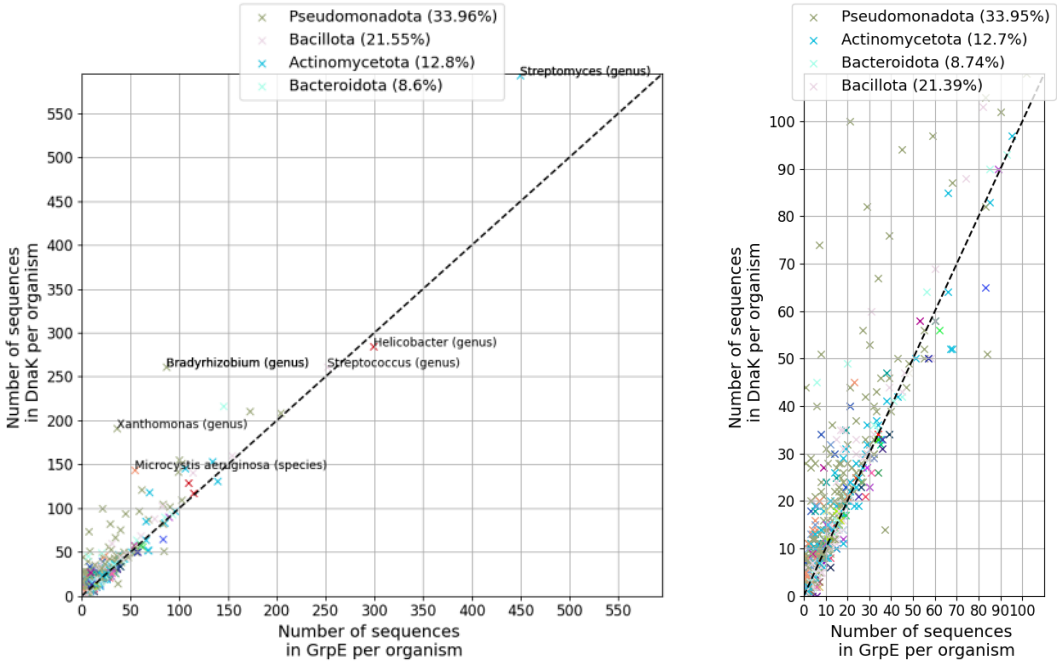
(C)



(D)

FIGURE 4.1: Two distinct dataset for the protein (A, B) DnaK and (C,D) GrpE. Phylum distribution with (A,C) the four biggest percentages of the data and the percentage of similarity with the reference sequence (B,D)

The data of GrpE and Dnak are from every kind of taxonomies. Their references proteins are selected such that Uniprot [29] informs us of an interaction between them: the Protein GrpE P09372 and the Chaperone protein DnaK P0A6Y8 (organism Escherichia coli (strain K12)). The distribution and the histogram of the different similarities between the sequences and the reference one are on Fig.4.1. The similarities show three distinct peaks for DnaK on Fig.4.1b and two peaks for GrpE on Fig.4.1d. For DnaK, the Pseudomonadota presents one group with an average of similarities with P0A6Y8 around 20% and another one around 70%. For GrpE, the pseudomonadota has a similarity with P09372 in average around 35%. The others taxonomies share the same peak than Bacillota, Actinomycetota and Bacteroidota (around 55% for DnaK and around 25% for GrpE).



(A) (B)

FIGURE 4.2: (A) Number of sequences per organism for GrpE and for Dnak (from the same that shared in Fig.3.1). The sequences with organisms defined as undefined or environmental sample have been removed. (B) Display of number of sequences/organism smaller than 100.

One phylum type contains several kind of organisms, and the data file can possess several sequences from the same organism. Before pairing extraction, a first analysis is done on the different numbers of sequences per organisms for GrpE and for DnaK. As the sequences from undefined or environmental samples are not technically attributed to an organism type, they are removed. The number of sequences per organism for the different protein types are shown on Fig.4.2. A general observation on Fig.4.2a reveals a quasi linear relation with some exceptions of organisms presenting deviation about 100-150 numbers of pairs (Streptomyces, Bradyrhizobium, and Xanthomonas). It is also observable that several kinds of organism present numerous sequences in both proteins (Streptomyces, Helicobacter, Streptococcus and Bradyrhizobium have more than 250 sequences/organism for GrpE and DnaK). An

examination of organisms with less than 100 sequences (see Fig.4.2b) displays a tendancy for Pseudomonadota type to have a higher numbers of sequences from DnaK.

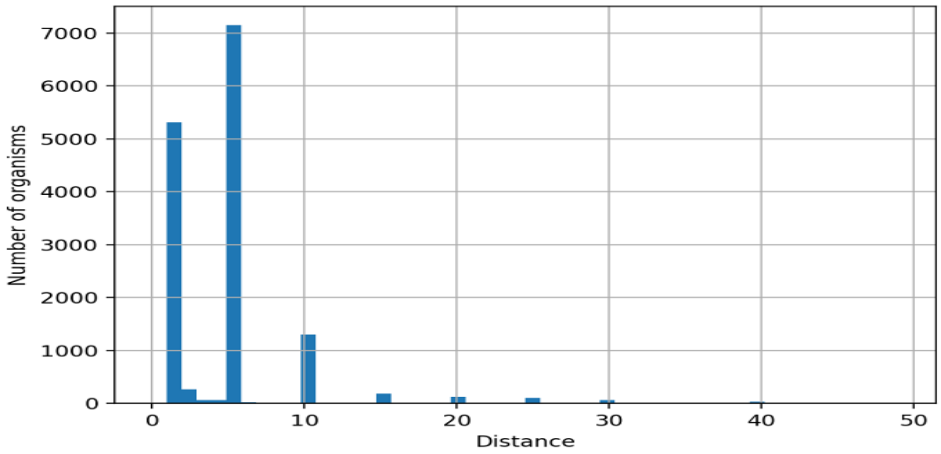
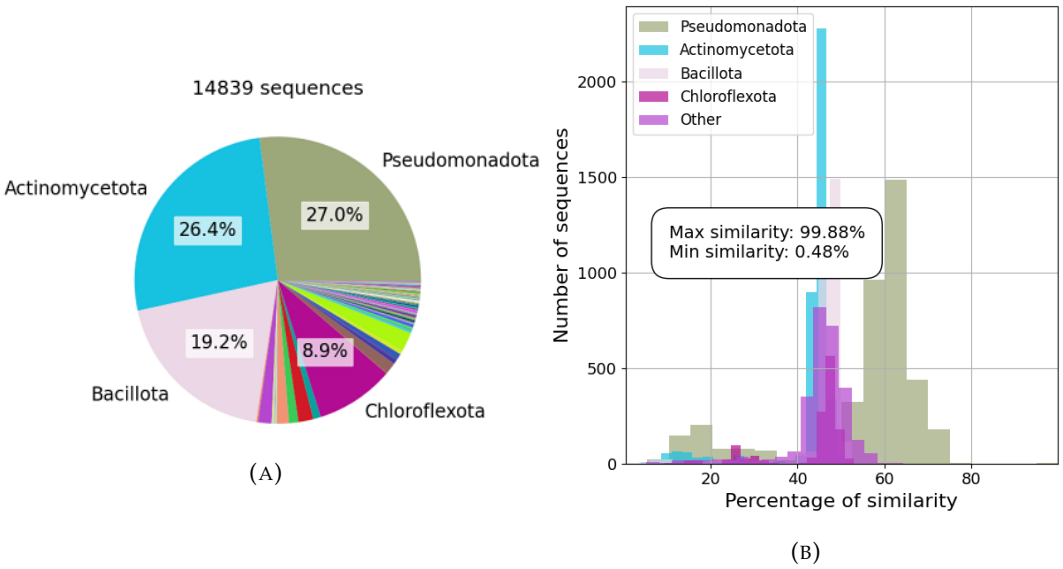


FIGURE 4.3: Distribution of numbers of GrpE-DnaK pairs according to the distance betweeen OLNs, or ORFs, numbers

After a first filtering by removing the sequences of unclassified or environmental samples, the pairs composition are obtained with a *MinDistance* of 50. An histogram of the distances according to these pairs is on on Fig.4.3. The distribution of similarities between the first sequence and the others is shown on Fig.4.4b and reveals two peaks. Pseudomonadota's sequences have a similarity around 62% (with the reference sequence) and all the other possible cases have a similar average around 47%. In order to reduce wrong pairing, the sequences pairs with less than 10% of similarity with the reference pair are removed. The final phylum distribution is found on Fig.4.4a.



(A) (B)

FIGURE 4.4: GrpE-DnaK (*MinDist* of 50): (B) The percentage of similarity with the reference sequences pair P09372-P0A6Y8 and (A) the phylum distribution after a filtering with $m_{\text{sim}} = 10\%$.

A final observation is done on the similarity between the pairs. The numbers of similar sequences with more than (X% identical amino acids) is computed for each sequence. Then the distribution of similar sequences between the four more present phylum in the dataset is computed and is found on Fig.4.5. This experiment reveals a similar number of identical amino acids between the pairs of each phylum ~ 50%.

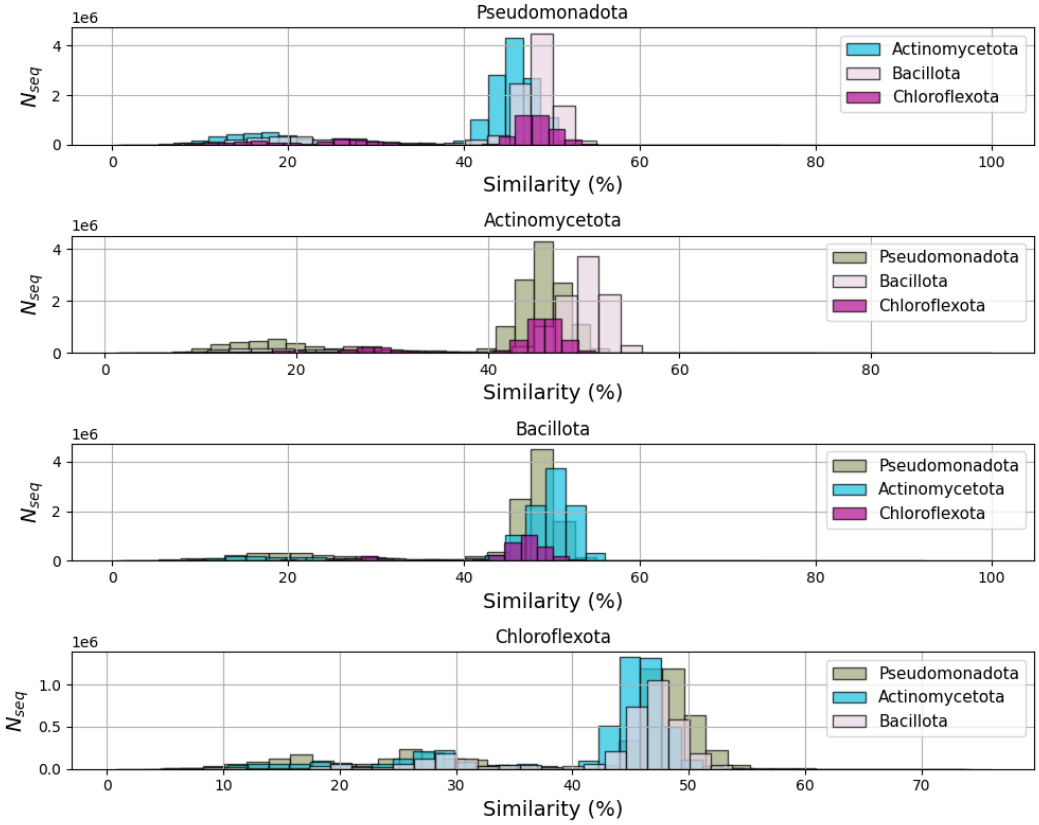


FIGURE 4.5: Distribution of the number of similar sequence with X% of same amino acids according the phylum.

4.1 Parameters of the models

The learning and test curves for different number of batchs, l_r, or μ are compared for the different dataset (Fig.4.6). The parameters choice for GrpE was complicated by the possible high gap between the learning and the test curve that could results from overfitting. A final choice of 1600 batchs has been applied to balance between a too small number of batchs (leading to a higher gap) and a too high number of batchs (leading to higher test curve for same number of epochs and a higher computational cost) (see Fig.4.6a). Note also how the gap between curves with 512 batchs and DnaK (on Fig.4.6d) isn't constant. Finally, by considering the others parameters, both GrpE and DnaK training seem to be more optimal with $l_r = 5e^{-4}$, and $\mu = 0.85$. Whereas, the bests parameters for the learning process of the pair GrpE-DnaK seems to correspond to the same Nbatchs and μ (see Fig.4.6j and 4.6l), the learning requests a smaller value of $l_r = 5e^{-5}$ (see Fig.4.6k). The final models were trained with these optimal parameters: 4000epochs, 1600 batchs, 0.85μ and Nesterov. $l_r = 1e^{-4}$ for the pairs and $l_r = 5e^{-4}$ for the protein alone. The different learning curves corresponding to these models are presented on Fig.4.7.

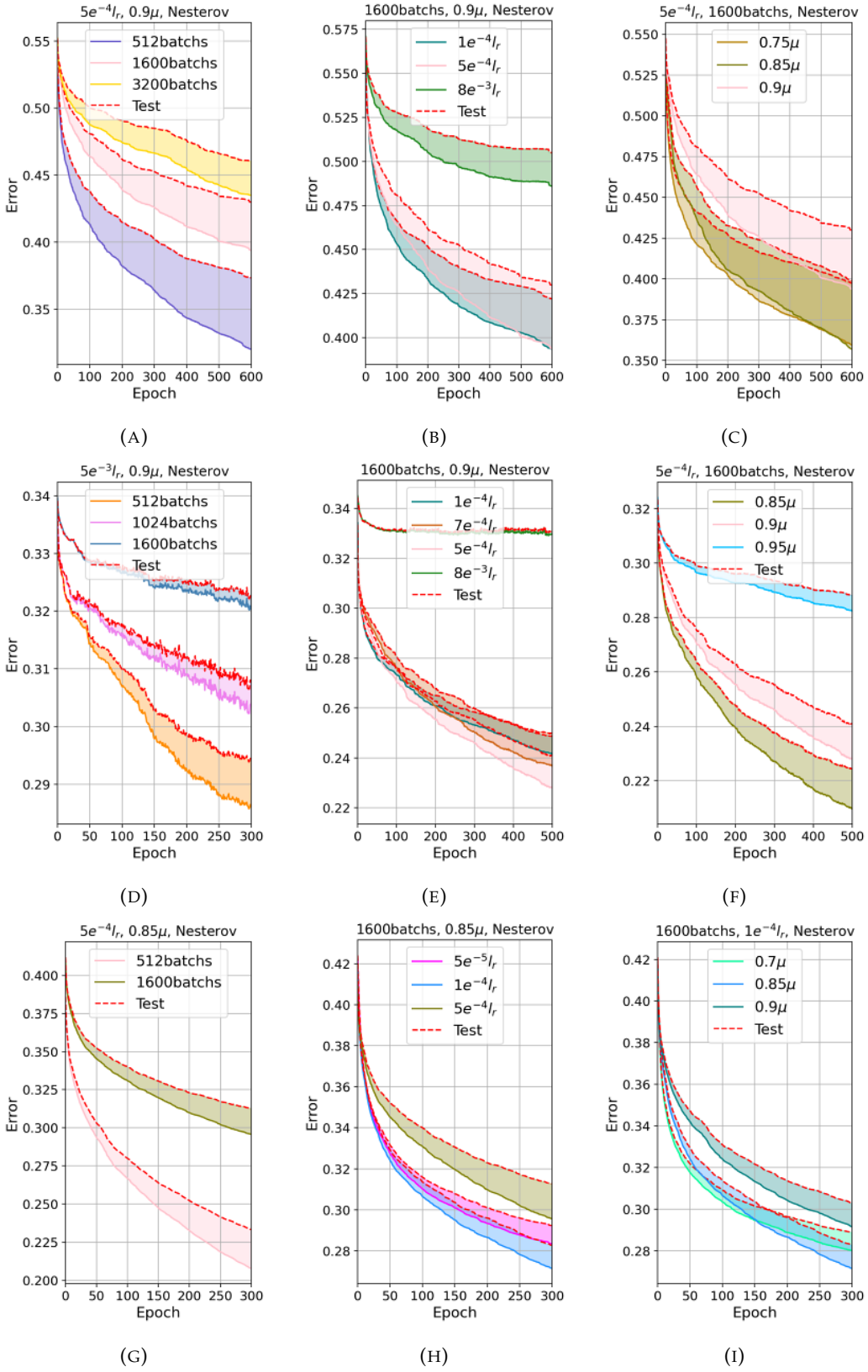
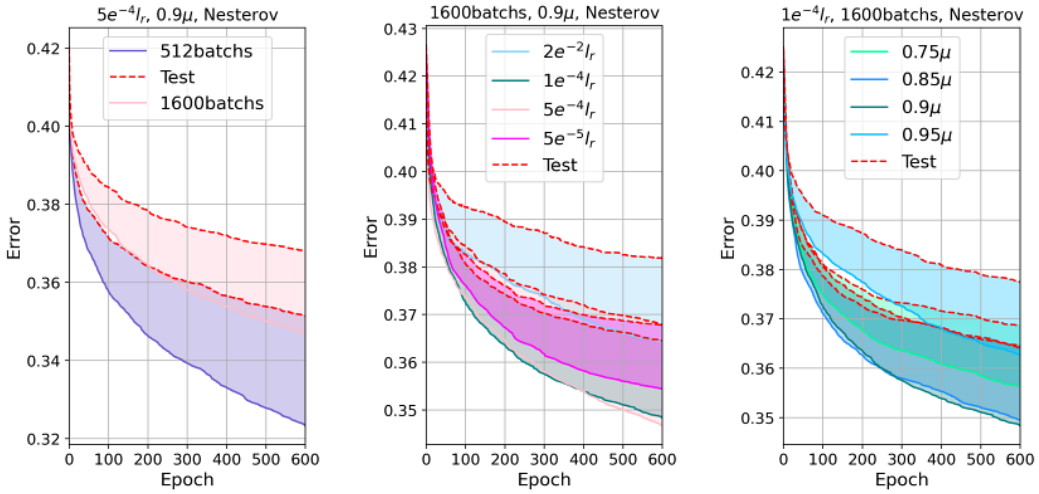
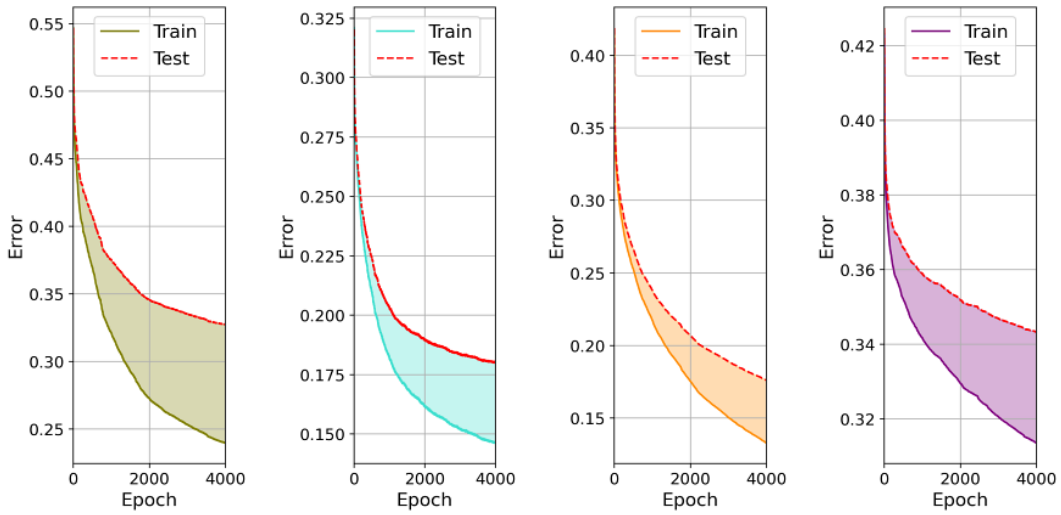


FIGURE 4.6: learning and testing curves for GrpE (A,B,C), DnaK (D,E,F), and GrpE-DnaK with linear model (G,H,I) and with cross linear model (J,K,L).



(J) (K) (L)

FIGURE 4.6: learning and testing curves for GrpE (A,B,C), DnaK (D,E,F), and GrpE-DnaK with linear model (G,H,I) and with cross linear model (J,K,L).

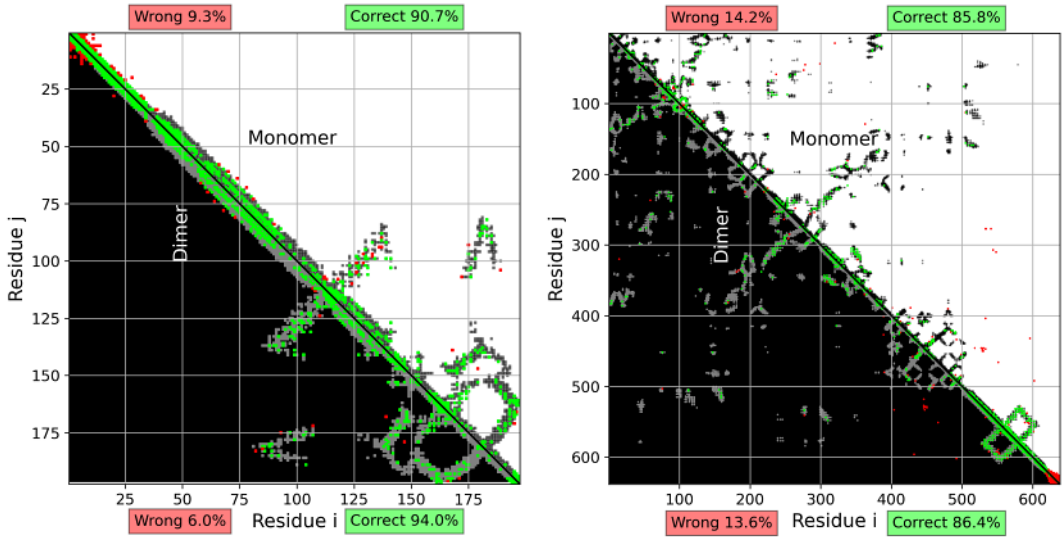


(A) (B) (C) (D)

FIGURE 4.7: learning and testing curves with 4000epochs, 1600 batchs, 0.85μ, Nesterov. $5e^{-4}l_r$ for (A)GrpE and (B)DnaK, $1e^{-4}l_r$ for GrpE-DnaK (A)linear and (B)cross-linear

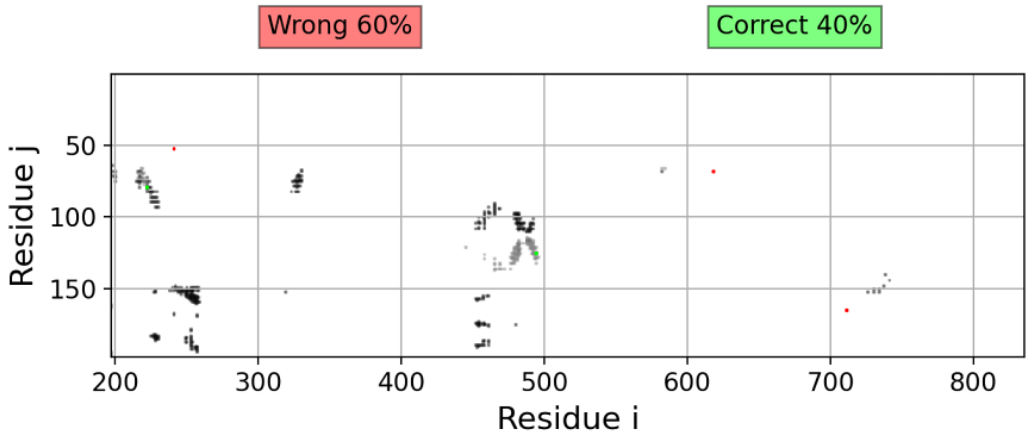
4.2 Contacts predictions

A first analysis is done by extracting the contacts and compare them to the ones of the monomers (or dimers) predicted by Alphafold. The 600 and 1000 contacts predictions for respectively GrpE and DnaK are on Fig.4.8 (a threshold distance of 8.5 Å). The predictions for the contacts between GrpE and DnaK were done thanks to the linear model (with all the amino acids during the training) or with the cross-linear model. The contact maps are found on Fig.4.9.

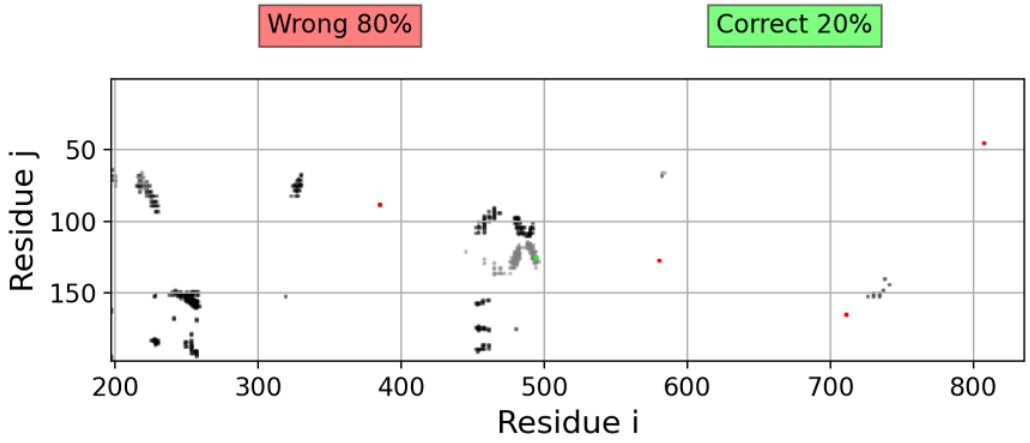


(A) (B)

FIGURE 4.8: 600/1000 contacts predictions for (A/B) GrpE/DnaK.



(A)



(B)

FIGURE 4.9: The 10/5 best scores contacts for GrpE-Dnak with (A) linear or (B) cross linear model.

For both proteins, most of the wrong predictions are still close to true ones. GrpE's predictions englobe almost all the surface predicted with Alphafold in comparison to the ones of DnaK that don't recover every spots. Additionnaly, the first 20 and last 20 amino acids for respectively GrpE and DnaK contact predictions seem to meet noise (look the huge quantity of red dots concentrate in these regions). Moreover, The consideration of the dimer improves the accuracy percentage of $\sim$ 3.3% for GrpE and of $\sim$ 0.6% for DnaK. The predicted contacts between these proteins seem very wrong (more than 50% errors). However, the same residus pair (125,494) shows up (*corresponding to the amino acids a_{125} for GrpE and a_{297}*).

A second analysis is done with delimited surfaces. It begins with the extraction of these areas with gaussian error maps (see Fig.4.10a & 4.10b) realised with the most optimal parameters (N_s, σ) for the observation of these different patterns.

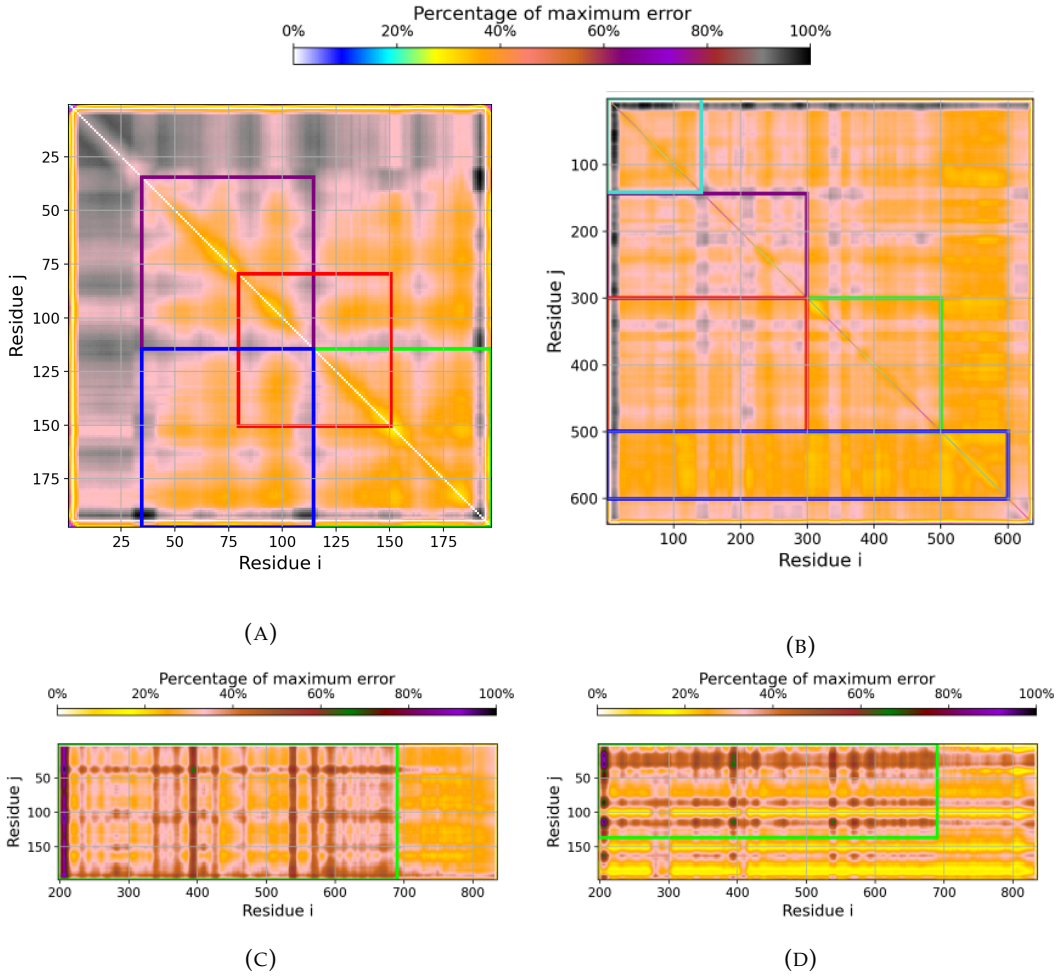
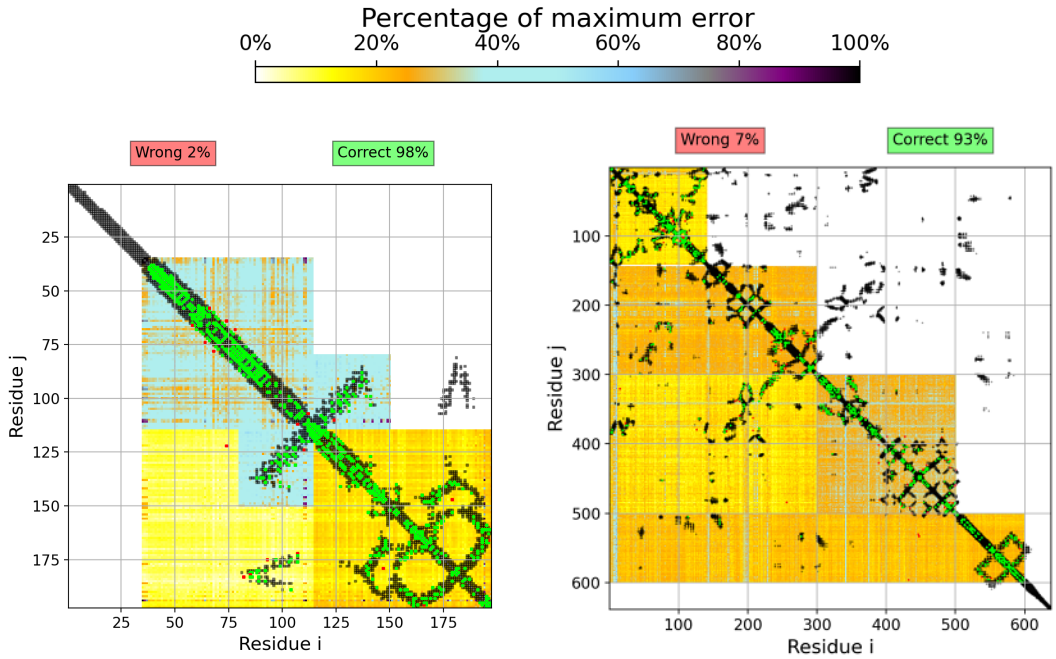


FIGURE 4.10: Noise variation for (N_s, σ) : (A)(6,6) in GrpE map, (B)(10,10) in DnaK map, and (C/D) (6,4) in GrpE-DnaK map with linear/linear-cross model.

Distinctive zones are visible for GrpE's map (see the orange marks $\sim$ 40% maximum error), it is more unclear for Dnak's map that contains these marks everywhere on the map. For this reason, almost the entire half map is going to be analysed. Predictions for these zones with reduction of errors (without Gaussian) are found on Fig.4.11. Since the regions with few noise variations are not considered, the region



(A) (B)

FIGURE 4.11: (A) 803 correct predictions against 19 wrong for GrpE and (B) 1245 correct predictions against 100 wrong for DnaK (C).

with the first 20 and last 20 amino acids (mentioned above) finally don't seem to be worth to be analysed.

As the regions are not of the same size and not all symmetric, different numbers of scores have been asked. Considering that N scores in a symmetric region is finally its half and that the diagonal predictions can be numerous, more scores can be extracted. For GrpE, 240 predictions are extracted in the three squares in the diagonal and 70 predictions are asked in the left-bottom corner. For DnaK, the scores extracted come from the 400 best scores in the two squares ($1 \leq (i,j) \geq 141$ and $300 \leq (i,j) \geq 501$), the 125 best scores in the two rectangular containing diagonal elements ($(1,144) \leq (i,j) \geq (299,299)$ and $(1,500) \leq (i,j) \geq (600,600)$), and the 70 scores in the left diagonal-external region. Take into account that the total number of contacts extracted could not be equal to the sum of the regions scores because regions overlap. The final results show an improvement in accuracy percentage on Fig.4.8 and Fig.4.11 of $\sim 4\%$ for GrpE and $\sim 6\%$ for DnaK.

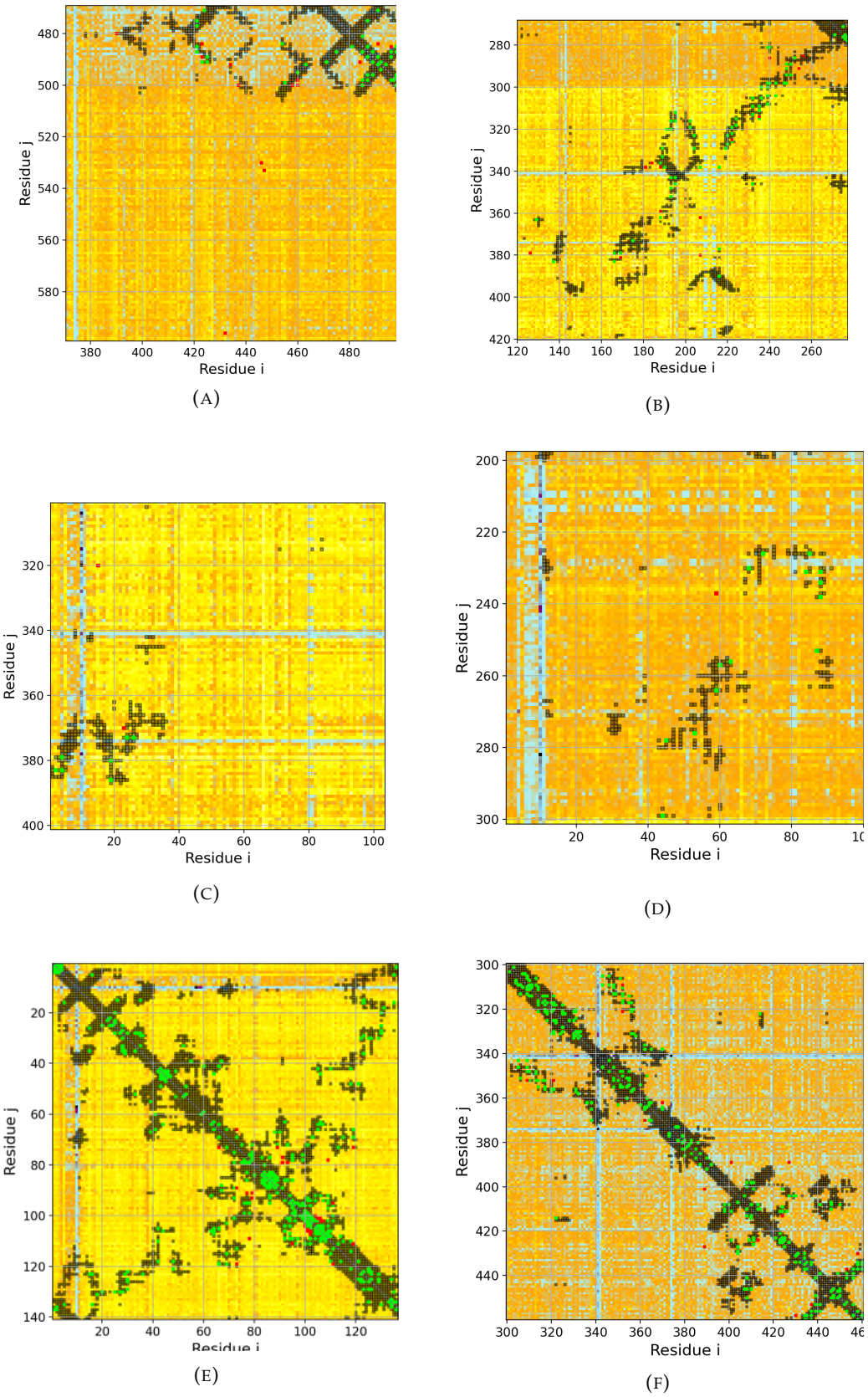
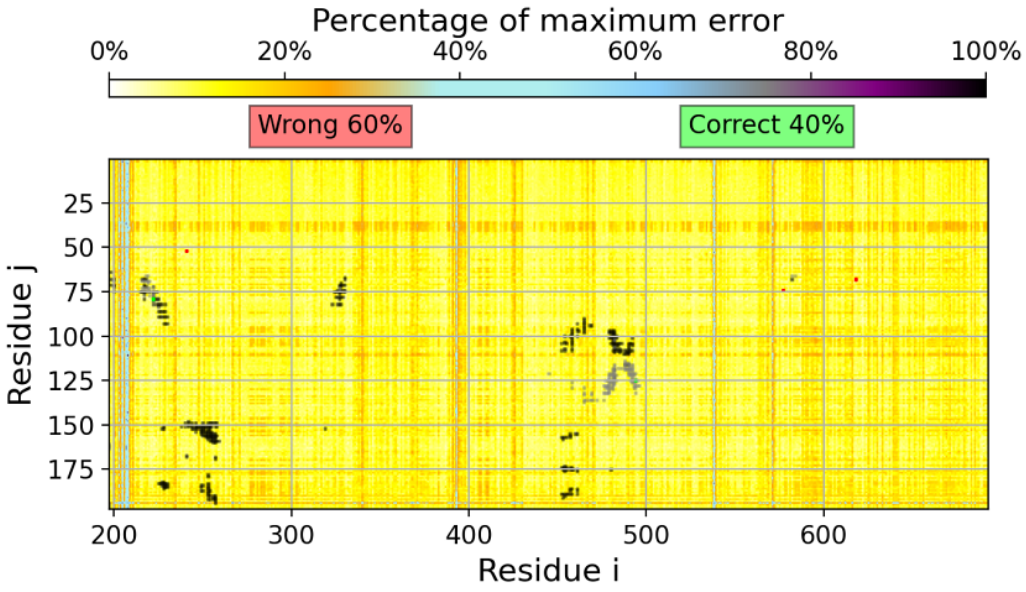
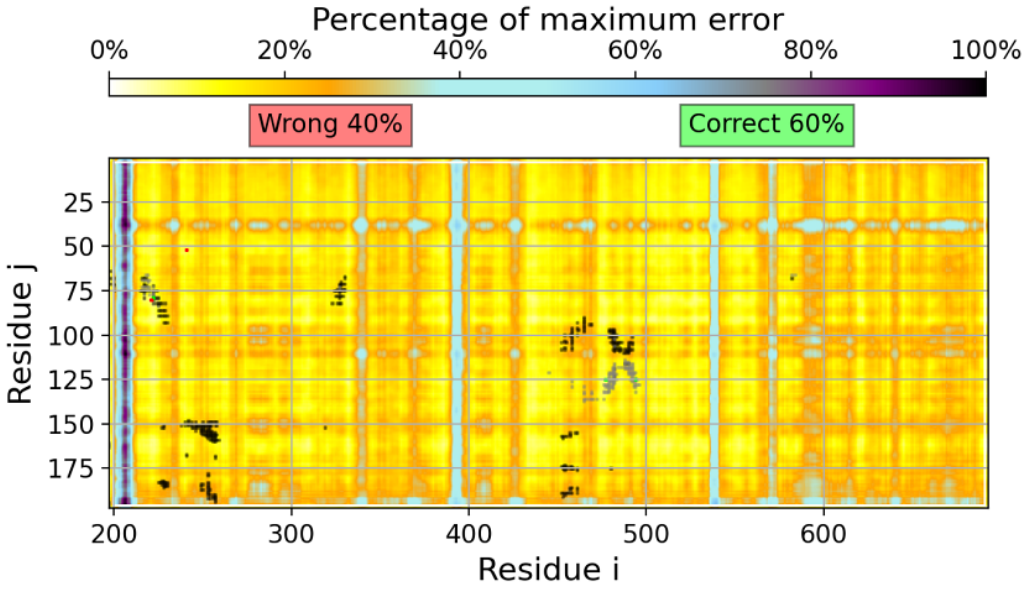


FIGURE 4.12: DnaK and crops from Fig.4.11b

Similarly specific regions are selected for the contacts predictions between GrpE and Dnak (with or without cross) (Fig.Fig.4.10c& Fig.4.10d). It can be observed that some areas are mainly composed with noise $\leq 25\%$, whereas others regions seem more interesting with noise variation. The resulting contacts with errors substraction (without or with Gauss) are found on Fig.4.13 for a linear model and on Fig.4.14 for a cross linear model. Note that crops of the map 4.13b are shown on Fig.4.15.



(A)

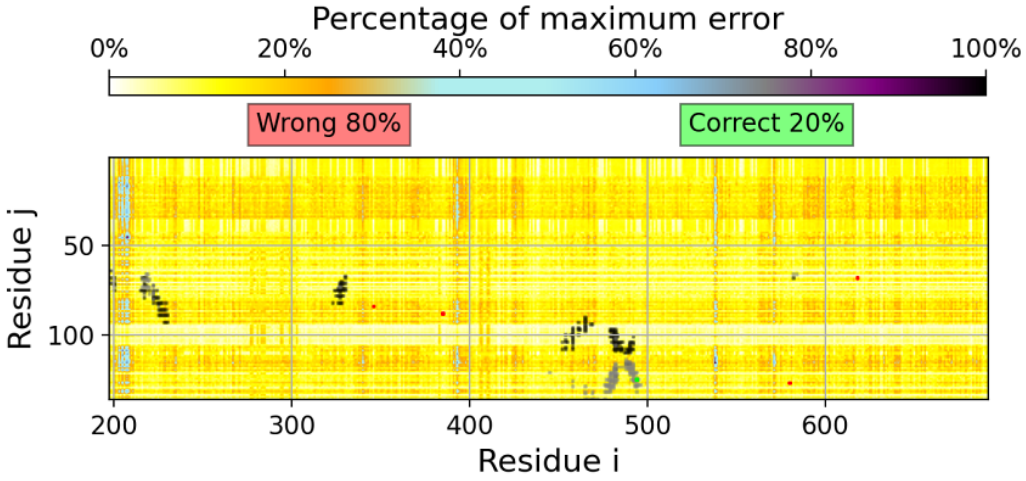


(B)

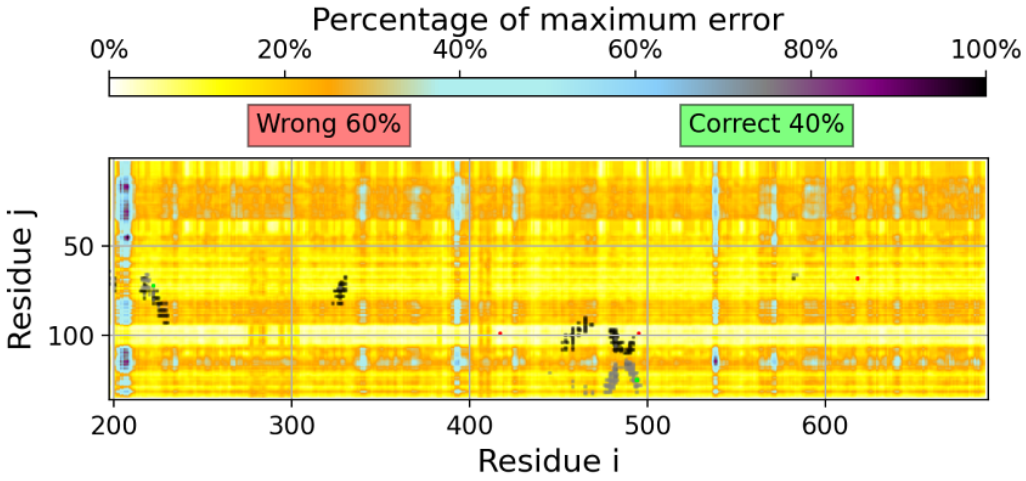
FIGURE 4.13: 5 GrpE-Dnak contact predictions map with the linear model (A)without and (B)with Gaussian error and $(N, \sigma) = (3, 2.5)$ for the region underlined on Fig.4.10c. Crops of regions are found on Fig.4.15.

The first five contacts predictions between GrpE and DnaK in all experiment are found in Tab.4.1 and the correct ones are shown with Alphafold on Fig.4.16. In every case, the highest score is the residus pair (125,494) (*corresponding to the amino*

acids a_{125} for GrpE and a_{297}) for DnaK which are (according to Alphafold) close to 3.25Å (see Fig.4.16). The Gaussian error consideration seems to reduce noisy predictions for both models. Both Fig.4.13a and Fig.4.14b highlight new correct contacts for respectively the residues pairs. Remark that the residues pairs (52,241) (*corresponding to (a_{52},a_{44})*) with linear model and the residues pairs (68,618) (*corresponding to (a_{68},a_{421})*) with cross-linear are still present despite the Gaussian errors.



(A)



(B)

FIGURE 4.14: 5 GrpE-Dnak contact predictions map with the linear mode for the region underlined on Fig.4.10c (A)without or (B)with Gaussian errors consideration and $(N, \sigma) = (2, 2)$ for the regions underlined on Fig.4.10d.

	linear				
A	$(a_{125}, a_{297})\star$	(a_{52}, a_{44})	$(a_{79}, a_{25})\star$	(a_{68}, a_{421})	(a_{74}, a_{380})
B	$(a_{125}, a_{297})\star$	$(a_{79}, a_{25})\star$	(a_{52}, a_{44})	$(a_{125}, a_{280})\star$	(a_{80}, a_{24})
	cross-linear				
A	$(a_{125}, a_{297})\star$	(a_{68}, a_{421})	(a_{84}, a_{149})	(a_{88}, a_{188})	(a_{127}, a_{383})
B	$(a_{125}, a_{297})\star$	(a_{99}, a_{220})	(a_{99}, a_{298})	(a_{68}, a_{421})	$(a_{72}, a_{25})\star$

TABLE 4.1: Predictions shown on Fig.4.13 and Fig.4.14between GrpE and DnaK according to two models and with an error correction (A) with Gaussian or (B) without. The stars indicate contacts $\leq$ 8.5Å.

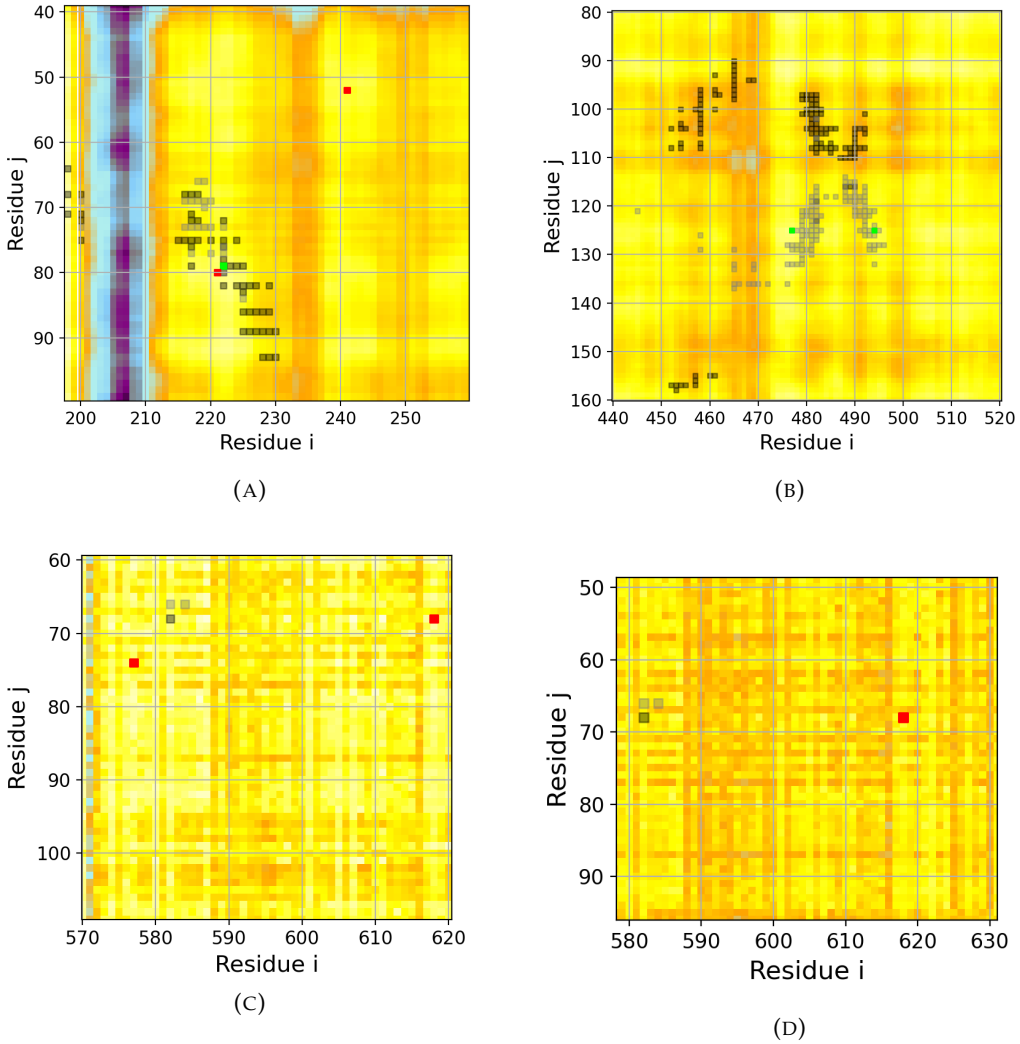
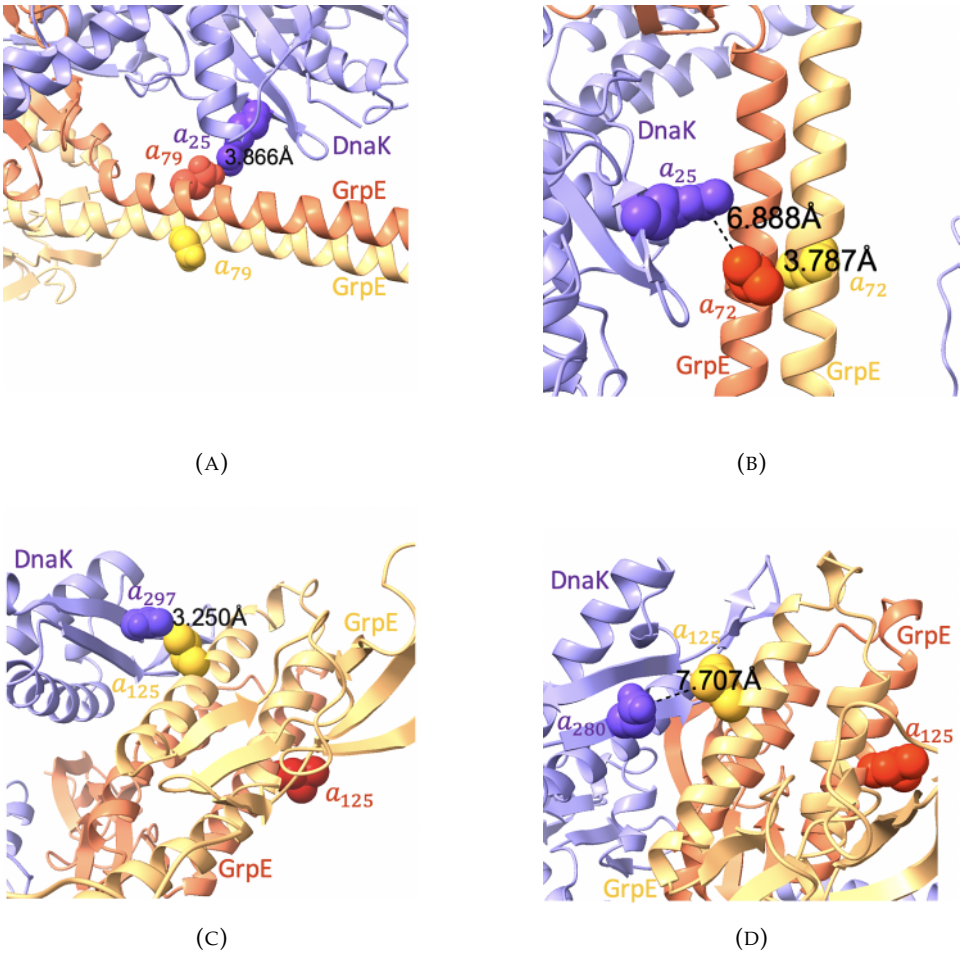


FIGURE 4.15: Region cropped on (A,B)Fig.4.13b and (C)Fig.4.13a.



(A) (B)

(C) (D)

FIGURE 4.16: The correct contacts ⋆ on Tab.4.1 presented with Alphafold [22].

4.3 Discussion

Since the numbers of GrpE and DnaK sequences are quite similar (57'619 for DnaK and 48'320 for GrpE) but DnaK lengths are three times bigger than GrpE length (Fig.4), it was probable that DnaK contacts extraction would be less accurate than for GrpE. In fact, the contacts map of GrpE seems more filled than the one of DnaK (Fig.4.8) but this is coherent with the difference in size and because the GrpE's map is only composed of one isolated contacts region external to the diagonal (against a high number for DnaK). Moreover, note also that it could have a relation with the average similarity between the sequences and the reference one that is smaller for GrpE than for DnaK (Fig. 4.1d & 4.1b). This resemblance could be the explanation of a faster decrease for DnaK dataset through the training and test (Fig.4.7).
The addition of dimer contacts reference decreasing the error percentage indicates that the amino acids sequence of a protein not only gives information about its structural contacts but also information about its potential contacts with the same sequence to form a dimer. The delimitation of regions with Gaussian on the error propagation helps in revealing uninteresting areas that don't seem to have numerous contacts outside of the diagonal part (Fig.4.10). In fact the improvement in accuracy looks mainly (thanks to these selections) avoiding a high quantity of noisy contacts for the first 20 and last 20 amino acids for respectively GrpE and DnaK. The

remaining errors that are still in majority close to true contacts (with 1 or 2 index difference) could probably be corrected with an averaged couplings (these ones obtained from different models) since it has been demonstrated in the previous section to increase the accuracy by reducing the noisy contacts.

The most probable contact between GrpE and DnaK being the same for any results (over the six presented) points out a strong force between these atoms (a_{125}, a_{297}). Furthermore, let's remind that the cross-linear model has learned the different weights between the amino acids that didn't belong to the same protein leading to the prediction of one protein thanks to another one. Thus, the discovery of the contact (a_{125}, a_{297}) as first prediction implies a co-information hidden between GrpE and DnaK propagated through all the homologous sequences during evolution. Furthermore, the use of selective regions with Gaussian error map propagation helped to find other contacts and one that was not detected by using the linear model. Additionally, the 3D representation of these contacts (Fig.4.16) indicates that the contacts found by using the Gaussian errors have a higher distance (6.888Å and 7.707Å) than the others (3.866Å and 3.250Å). This could suggest that the high sensibility of the noise in the contacts predictions avoid the extraction of the weaker contacts that are masked by the noisy ones.

Chapter 5

Hyou1 (Grp170) and immunoglobulin heavy-chain binding protein (BiP)

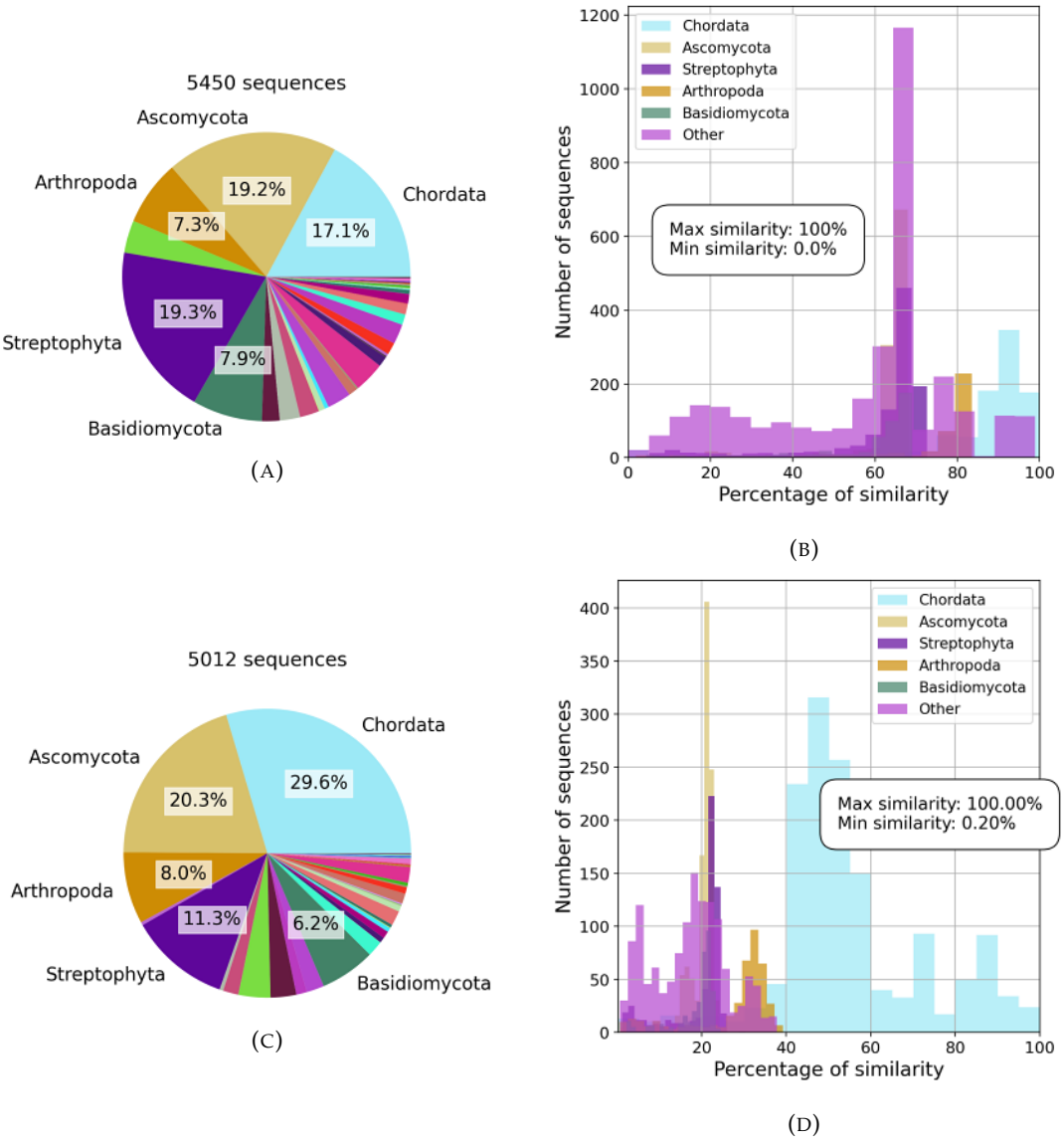
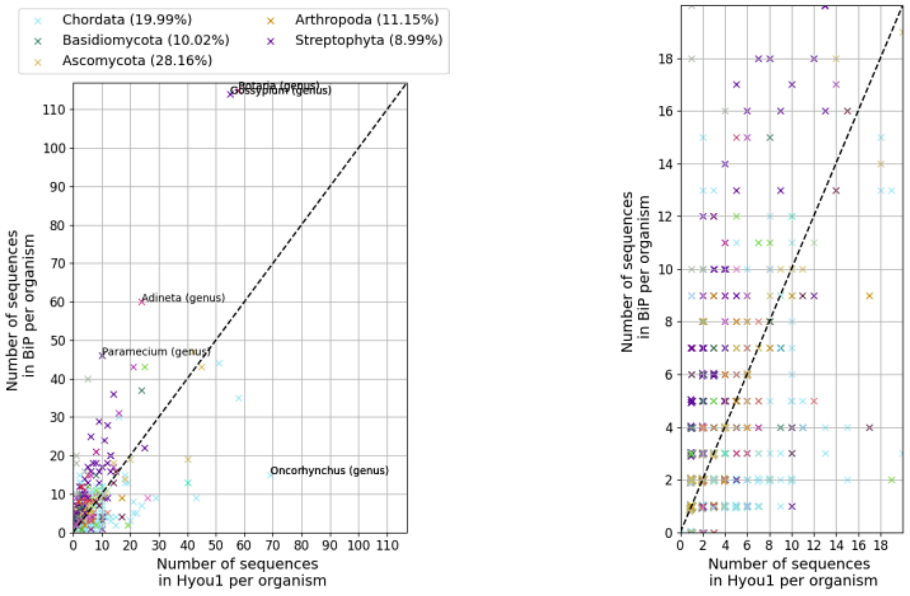


FIGURE 5.1: Phylum distribution with (A,C) the four biggest percentage of the data and the percentage of similarity with the reference sequence (B,D). (A,B) BiP, (C,D) Hyou1.

The data of Hyou1 and BiP are from every kind of taxonomies. Their references proteins are selected such that Uniprot [29] informs us of an interaction between them: the Protein Hypoxia up-regulated protein 1 Q9Y4L1 and the Endoplasmic reticulum chaperone BiP P11021 (organism Homo sapiens (Human)). The different phylum distribution of these sequences and their similarities with the reference sequences are find on Fig.5.1. The histogram about the BiP dataset presents two different behaviours: The chordata sequences seem to have a varied spectrum of similarity whereas the other phylum shared a similarity average with the reference sequence around 20%. Since the alignment of BiP results in a high quantity of sequences with more gaps than characters, a preprocessing with a threshold of 80% gaps authorised has been fixed.

As for the previous chapter, the idea being to determine the contact between Hyou1 and BiP, the numbers of sequences per organisms according to the proteins are shown on Fig.5.2a. Observe the disequilibrium of number of sequence for Streptophyta and Chordate on Fig.5.2b and how the quasy-linearity is less visible than with the prokaryotic ones on Fig.4.2.



(A) (B)

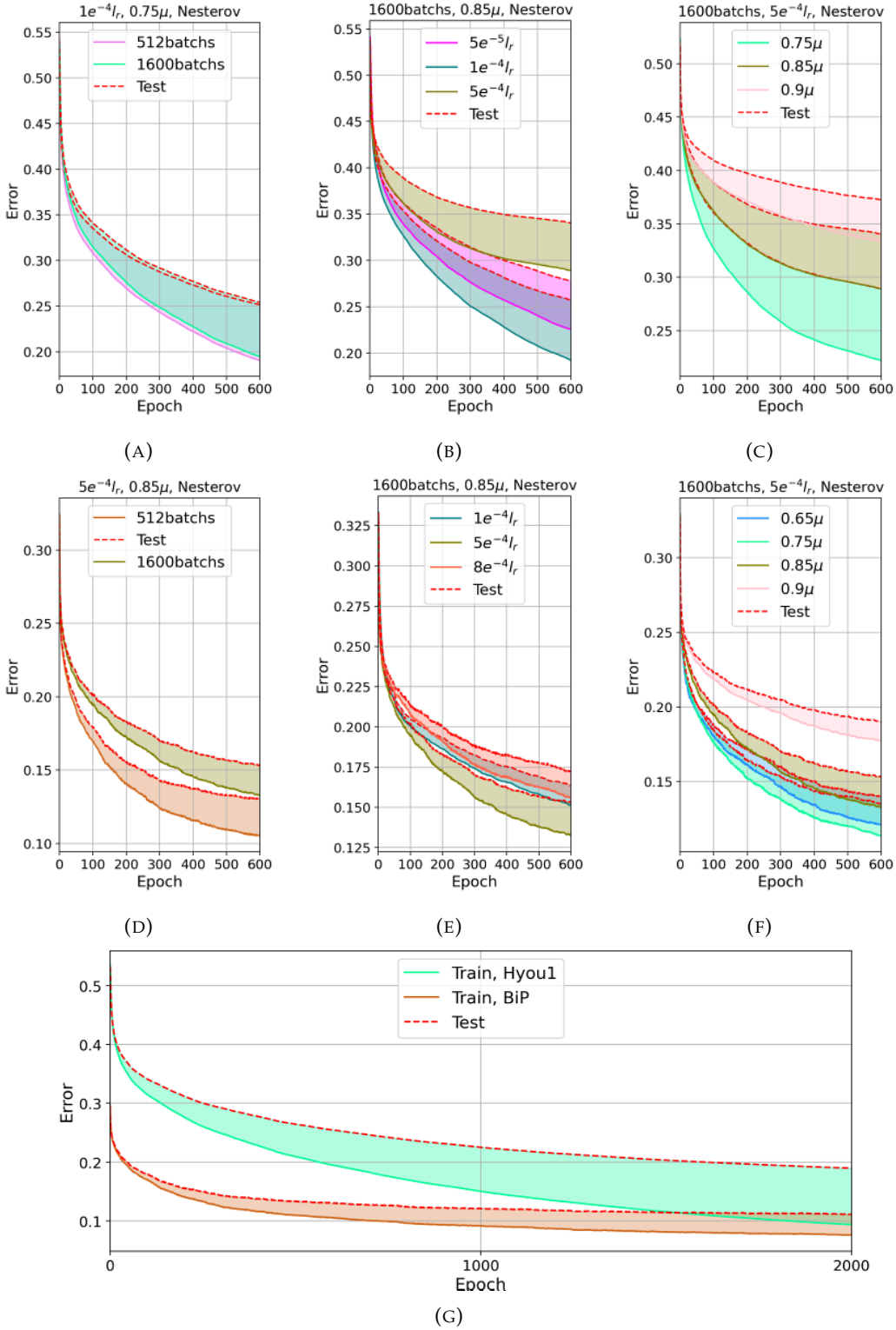
FIGURE 5.2: (A) Number of sequences per organism for Hyou1 and for BiP. The sequences with organisms defined as undefined or environmental sample have been removed. (B) Display of number of sequences/organism smaller than 20.

In addition to the pairing problem that was made randomly (because no OLNs or ORFs as in prokaryote), the computational task was impossible for the current Graphics Processing Unit (GPU) [1] used to train the models. Indeed, the resulting sequences have a length of 1653 amino acids ($M_{\text{Q9Y4L1}} + M_{\text{P11021}}$); what is more than the double of the GrpE-DnaK sequences length. However, the contacts predictions of the protein BiP and the protein Hyou1 can be extracted.

[1] NVIDIA GeForce RTX 4090, 24564MiB memory usage

5.1 Parameters of the models

Different parameters (batchs, learning rate l_r, and momentum μ) are compared and the learning and testing curves are found on Fig.5.3.



(A) (B) (C)

(D) (E) (F)

(G)

FIGURE 5.3: learning and testing curves for Hyou1 (A,B,C) and BiP (D,E,F). The most optimal curves with 3000 and 2000 epochs (G).

The learning rate comparisons on Fig.5.3b and Fig.5.3e show clearly a preference for a smaller value for Hyou1 around $1e^{-4}$, whereas this value is no the best for BiP training that should be more optimal with $l_r = 5e^{-4}$. Additionally, since the curve training of BiP sequences seem to decrease faster around smaller values than it was the case with previous dataset, it was decided to take a moment of 0.85μ. This choice should slow down the training and maybe help the model to have a bigger view of the weights between the amino acids (see Fig.5.3f). The final chosen parameters are $1e^{-4}l_r, 0.75\mu$, 1600batchs and 3000epochs for Hyou1, and $5e^{-4}l_r, 0.85\mu$, 512batchs and 2000epochs for BiP.

5.2 Contacts predictions

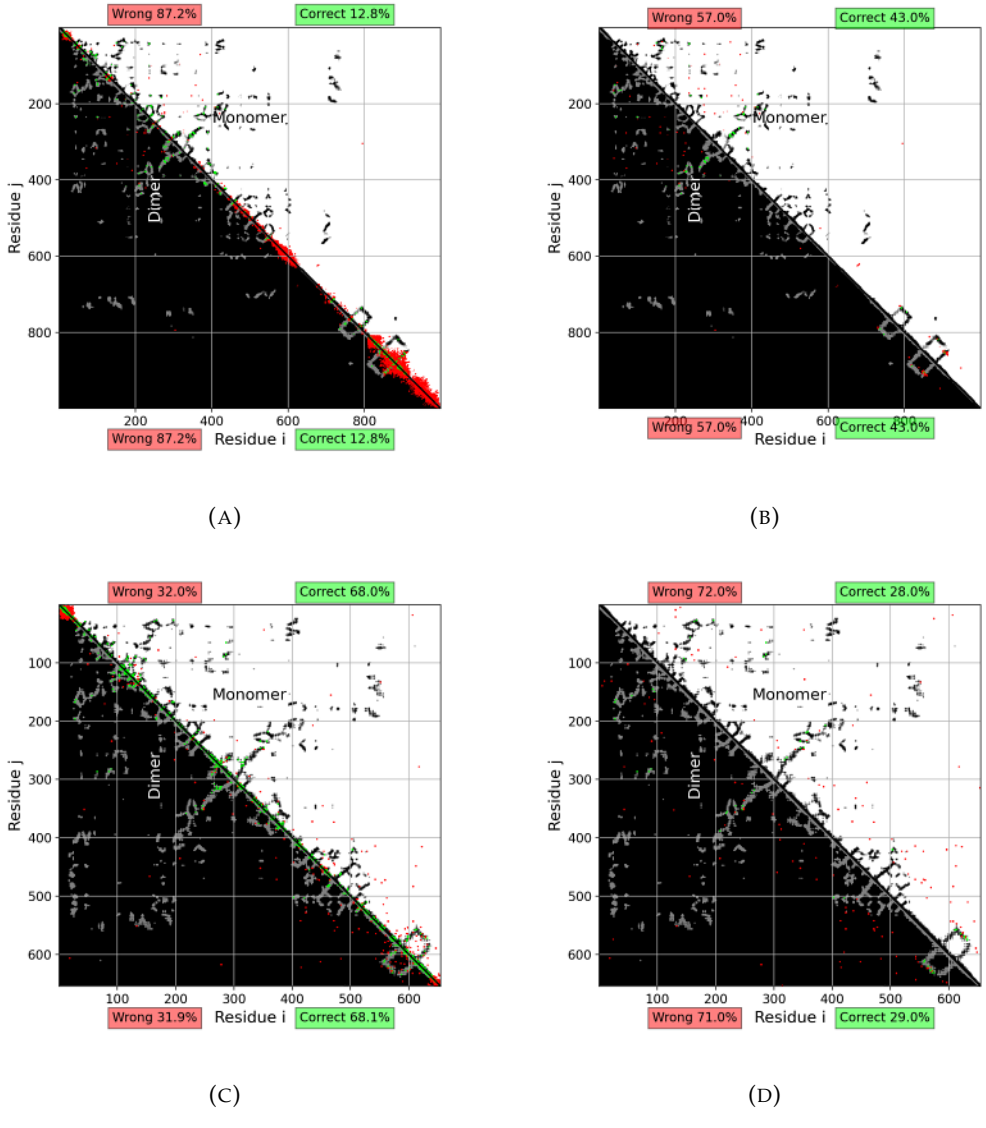
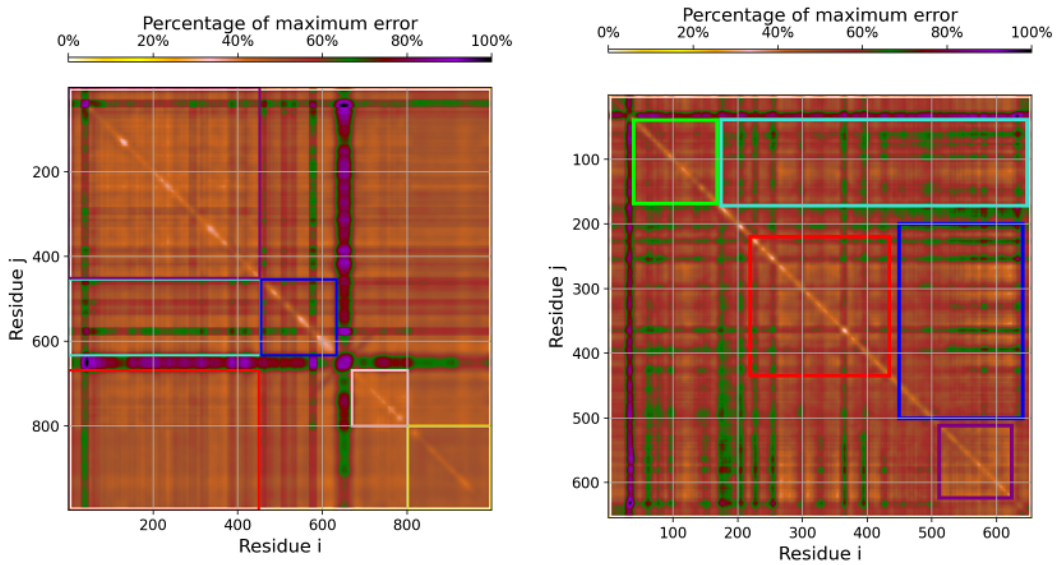


FIGURE 5.4: N Contacts predictions for every elements except the ones in the diagonal $\pm D$ region for (A,B) Hyou1 and (C,D) BiP. With (N, D) : A.(2000,0), B.(100,50), C.(800,0), D.(100,50).

A first analysis is done by extracting the contacts and compare them to the ones of the monomers (or dimers) predicted by Alphafold. The 800 and 2000 contacts predictions for respectively BiP and Hyou1 are on Fig.5.4 (a threshold distance of 8.5 Å). Since the majority of predictions seem to be trapped in the diagonal's region (Fig.5.4a&5.4c, another observation is done by avoiding the contact at a distance ± 50 of the diagonal. However, this modification didn't seem to improve the external predictions. Additionally, it seems that a high quantity of noise is present in the region $i, j \leq 400$ for Hyou1 and almost everywhere for BiP. In order to compensate this noise, a second analysis is done with delimited surfaces. It begins with the extraction of the area with gaussian error maps (see Fig.5.5a & 5.5b).



(A) (B)

FIGURE 5.5: Noise variation for (N_s, σ) : A.(10,10) in Hyou1 map and B.(6,6) in BiP map.

The maps are principally recovered of errors between 40% and 60% with some extra green lines $\sim$ 70% and the diagonal part $\sim$ 35%. The delimitation are applied around the different region of distinguishable brown colors. Then different numbers of predictions were extracted from these different areas: Hyou1 with 200 ($\times$ 2) for the biggest symmetrical square, 50 ($\times$ 2) for the smaller symmetrical squares, 70 ($\times$ 2) for the last one in the bottom right-hand corner, and 15 for the rectangular shape regions (external to the diagonal). BiP with 75 ($\times$ 2) for the biggest symmetrical square, 50 for the other symmetrical squares, and 15 for the rectangular shape regions (external to the diagonal). The final predictions obtained with errors Gaussian or not are found on Fig.5.6. Some improvement in the diagonal regions are observable for BiP but, as for Hyou1, the concentrations of contacts in one or two regions are still mainly impacting the predictions. These concentration are visible with the crops found on Fig.5.7 and 5.8. Additionally, no improvement was bring out in the diagonal-external regions.

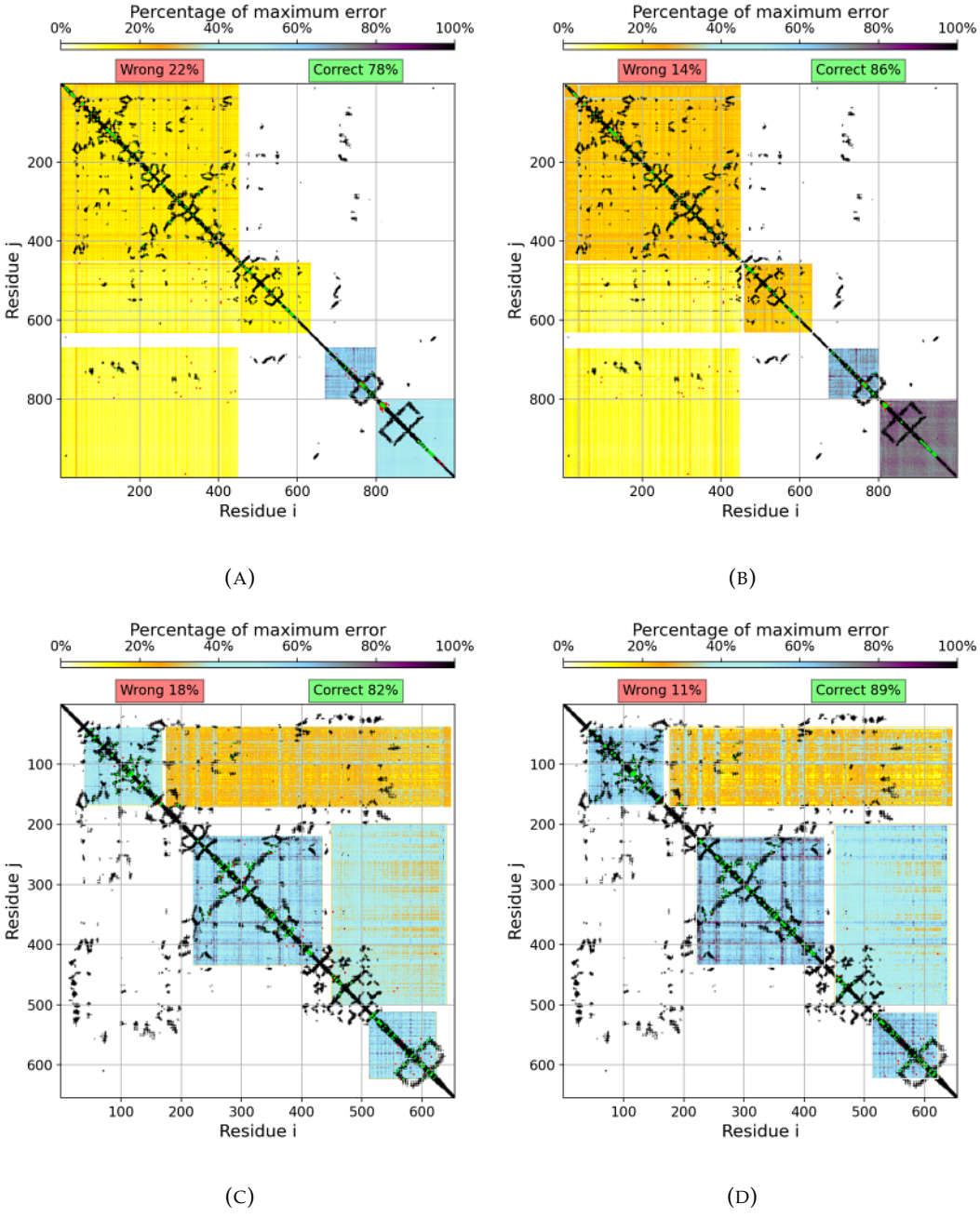
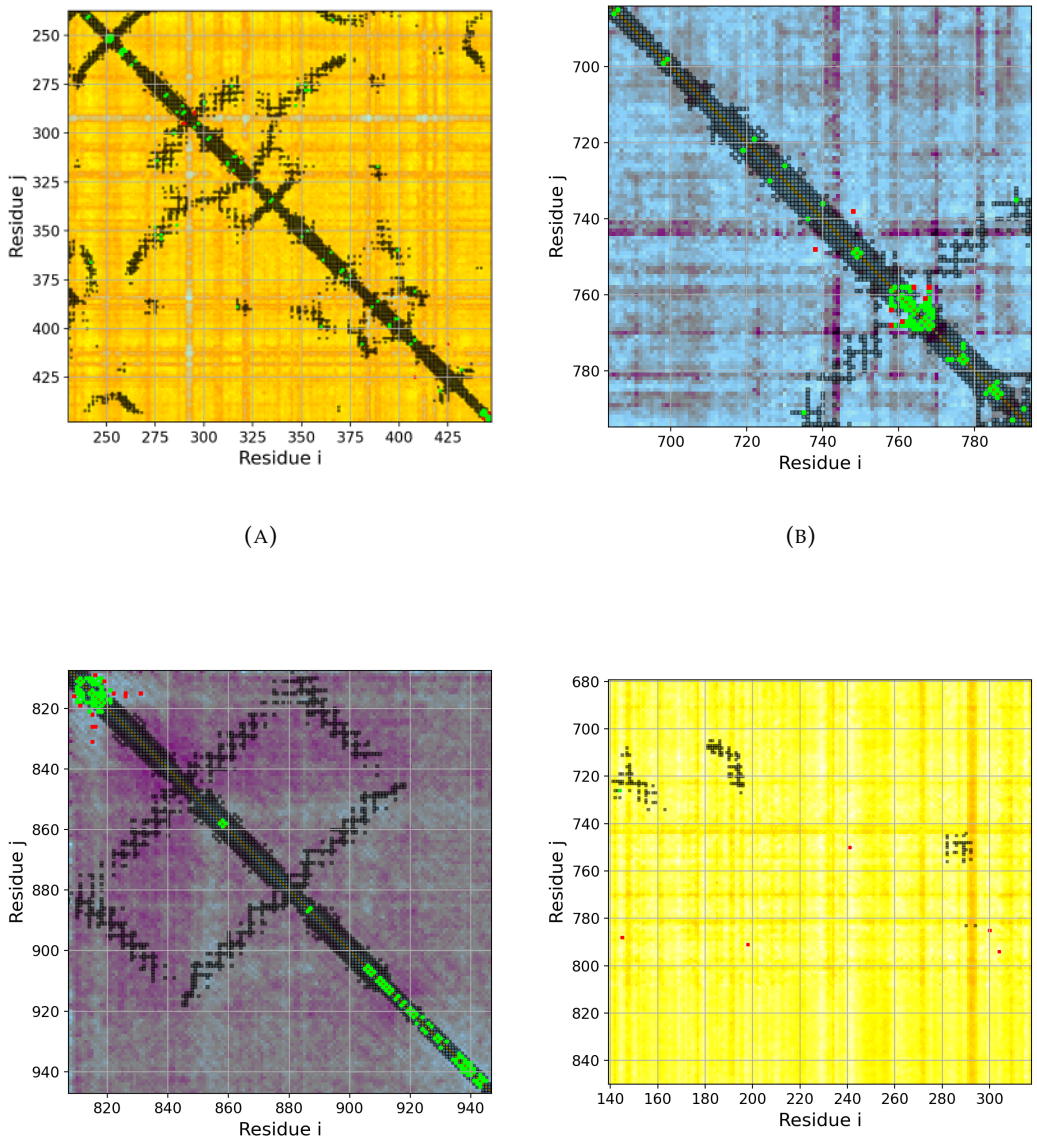


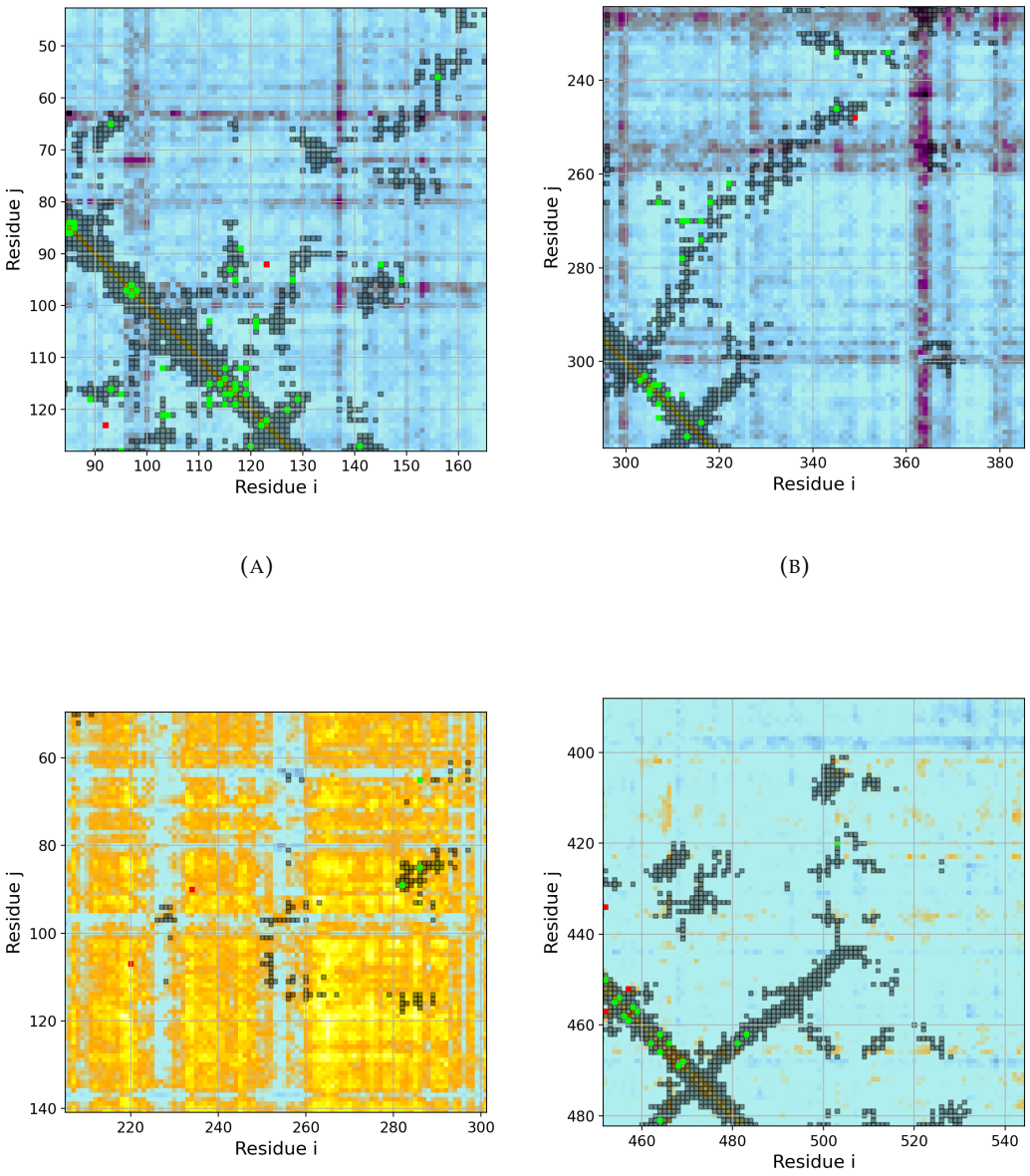
FIGURE 5.6: (A,B) Hyou1 and (C,D) BiP contacts predictions (A,C) without and (B,D) with Gaussian errors and $(N, \sigma) = (2, 0.5)$.



(A) (B)

(C) (D)

FIGURE 5.7: Hyou1 crops from Fig.5.6b.



(A) (B)

(C) (D)

FIGURE 5.8: BiP crops from Fig.5.6d.

5.3 Discussion

The poor contact predictions for BiP and Hyou1 are coherent with the fact that the number of sequences used for the training was not enough. Indeed, BiP dataset contains 10 times less sequences than DnaK dataset (5'450 vs 57'619, see Fig.4.1a & 4.1a) but with almost the same number of amino acids (654 vs 638, see Fig.4). This is even worst with Hyou1 dataset that has a similar amount of sequences (5'010, see Fig.5.1c) but with $\sim$ 1.5 times more amino acids (999, see Fig.4). This can explain why the contact map was even worse for Hyou1 than for Bip. Moreover the learning curves fell faster to a smaller error percentage (Fig.5.3) because it was strongly possible that the sequences difference and information was not sufficient. Regarding the contacts maps and the cluster formed in some regions in the diagonal, the model mostly learned to find some amino acids next to others in the sequence. In this case no improvement was possible with region's selection because no contact were learning in the region too outside from the diagonal. However, the initiation of contacts distant about 40 indices from the diagonal could indicate that the contacts predictions should work for a model learned with a larger dataset.

Chapter 6

Conclusion

Different advantages and disadvantages have been discovered thanks to the different datasets overlapping the prokaryote and eukaryote domains. The model using DCA and pseudolikelihood has revealed itself to be useful in predicting contacts, and also in other fields as the amino acids predictions and the phosphorylation identification. Furthermore, strategies involving noise and Gaussian interpretations appeared to be useful in the identification of areas composed of contacts between amino acids that are not neighbors from one to another. This also allowed to focus the predictions in these regions and to discover other contacts that were hidden by the strong contacts present in the neighboring or created by the noise.

However, the model has its limits and it appeared that accurate contacts predictions are possible only if the number of sequences for the model training is enough, considering the number of amino acids per sequence (BiP, Hyou1). Note also that it appeared that the similarity between the sequences of the dataset and the first sequence is also determining. Additionally the phylum identification had a poor accuracy of less than 30% (DnaJ). This could be the cause of a disparity in taxonomy distribution with a type that is more abundant than the others. Moreover, only results from linear or cross-linear models were presented since the non-linear couplings extraction asked a high computation cost. This would require consecutive forloops during the Ising computation with a 3 dimensional matrix of size 1365^3 (DnaJ) in the best case and of size $20'979^3$ (BiP) in the worst case.

On the other hand, when the condition on the dataset size and the similarity average with the first sequence were satisfied, the contacts predictions have been seen to reach an accuracy close to 94% for 600 contacts between 199 amino acids. This was done with GrpE dataset containing a sequences numbers more than 245 times the sequences lengths, and an average similarity with the first sequence around 30%. Moreover, the comparisons of the contacts predictions between monomer and dimer allow to identify the connection inside two same sequences and exhibit the influence of evolution and ability of an amino acids chain to already know where to bind to form a dimer. Additionally, it seems that the knowledge of a protein can be predicted with another protein type if they have the possibility to form a pair. This suggests that the life expansion has engraved complementary codes for proteins that can bind together. The contacts predictions between these pairs have revealed amino acids in proximity with a distance around 3Å according to Alphafold (GrpE, DnaK).

Finally, I address my special thanks to the professor Paolo De Los Rios for having supervised my master thesis, and given me access to new tools and knowledge. It was also a great pleasure to get to know other people from the laboratory and to learn from their experience during presentations and discussions.

Appendix A

Probabilities computation details

($*^1$) : The probability to have the sequence **a** (with $(a_1, ..., a_{j-1}, a_{j+1}, .., a_M)$ fixed to the values $(a_1^l, ..., a_{j-1}^l, a_{j+1}^l, .., a_M^l)$ and a_j free) is given by the summation of probabilities of every possible sequences with $a_j \in [1, K]$

($*^2$) : The probability to have the sequence **a** with $(a_1, ..., a_{j-1}, a_j, a_{j+1}, .., a_M)$ fixed to the values $(a_1^l, ..., a_{j-1}^l, a_j^l, a_{j+1}^l, .., a_M^l)$ can be expressed with Eq.(1.1):

$$P(a_j = a_j^l, \mathbf{a}_{/j} = \mathbf{a}_{/j}^l) = \frac{1}{Z} \exp\left(\sum_{\substack{m=1 \\ m \neq j}}^{M} \mathbf{h_m}(a_m^l) + \sum_{\substack{m=1 \\ m,n \neq j}}^{M-1} \sum_{n=m+1}^{M} \mathbf{J_{mn}}(a_m^l, a_n^l) \right)$$
$$\times \exp\left(\mathbf{h_j}(a_j^l) + \sum_{m=1}^{j-1} \mathbf{J_{mj}}(a_m^l, a_j^l) + \sum_{n=j+1}^{M} \mathbf{J_{jn}}(a_j^l, a_n^l) \right)$$

which becomes with the symmetric propriety of **J**:

$$P(a_j = a_j^l, \mathbf{a}_{/j} = \mathbf{a}_{/j}^l) = \frac{1}{Z} \exp\left(\sum_{\substack{m=1 \\ m \neq j}}^{M} \mathbf{h_m}(a_m^l) + \sum_{\substack{m=1 \\ m,n \neq j}}^{M-1} \sum_{n=m+1}^{M} \mathbf{J_{mn}}(a_m^l, a_n^l) \right)$$
$$\times \exp\left(\mathbf{h_j}(a_j^l) + \sum_{\substack{n=1 \\ n \neq j}}^{M} \mathbf{J_{jn}}(a_j^l, a_n^l) \right)$$

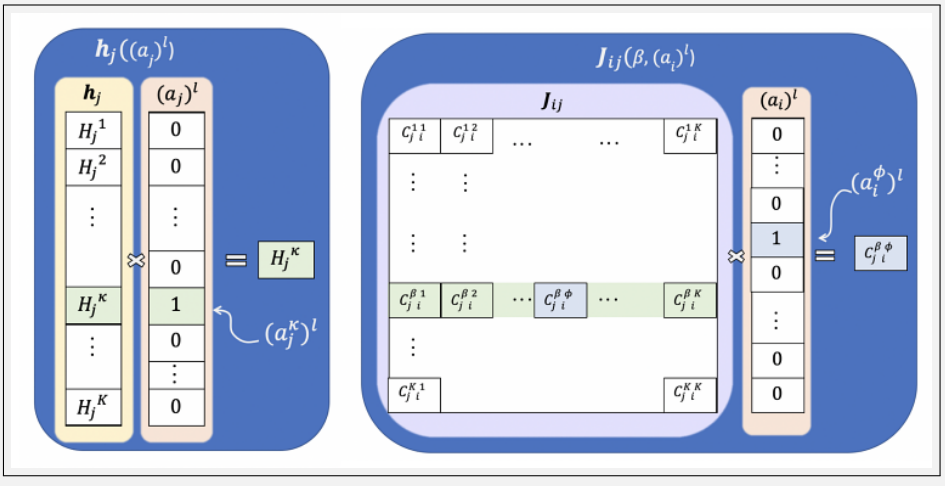
($*^3$) : The probability to have the sequence $\mathbf{a}$ with $(a_1, ..., a_{j-1}, a_j, a_{j+1}, .., a_M)$ fixed to the values $(a_1^l, ..., a_{j-1}^l, \beta, a_{j+1}^l, .., a_M^l)$ can be expressed with Eq.(1.1):

$$P(a_j = a_j^l, \mathbf{a}_{/j} = \mathbf{a}_{/j}^l) = \frac{1}{Z} \exp\left(\sum_{\substack{m=1 \\ m\neq j}}^{M} \mathbf{h_m}(a_m^l) + \sum_{m=1}^{M-1} \sum_{\substack{n=m+1 \\ m,n\neq j}}^{M} \mathbf{J_{mn}}(a_m^l, a_n^l) \right)$$
$$\times \exp\left(\mathbf{h_j}(\beta) + \sum_{m=1}^{j-1} \mathbf{J_{mj}}(a_m^l, \beta) + \sum_{n=j+1}^{M} \mathbf{J_{jn}}(\beta, a_n^l) \right)$$

which becomes with the symmetric propriety of $\mathbf{J}$:

$$P(a_j = a_j^l, \mathbf{a}_{/j} = \mathbf{a}_{/j}^l) = \frac{1}{Z} \exp\left(\sum_{\substack{m=1 \\ m\neq j}}^{M} \mathbf{h_m}(a_m^l) + \sum_{m=1}^{M-1} \sum_{\substack{n=m+1 \\ m,n\neq j}}^{M} \mathbf{J_{mn}}(a_m^l, a_n^l) \right)$$
$$\times \exp\left(\mathbf{h_j}(\beta) + \sum_{\substack{n=1 \\ n\neq j}}^{M} \mathbf{J_{jn}}(\beta, a_n^l) \right)$$

($*^4$) Since $\mathbf{h_j}$ is a vector of size K with elements H_j^β ($\beta \in [1, K]$), the elements $\mathbf{h}_j(\beta)$ can be expressed as H_j^β. Additionally, since $(a_j)^l$ is an one hot encoder vector of size K, the element formulation $\mathbf{h}_j((a_j)^l)$ can be expressed as $\sum_{\kappa=1}^{K} H_j^\kappa \cdot (a_j^\kappa)^l$. In the other hand, since $\mathbf{J}_{ij}$ is a matrix of size $K \times K$ with elements $C_{ji}^{\beta\alpha}$, the expression $\mathbf{J}_{ji}(\beta, (a_i)^l)$ can been seen as $\sum_{\phi=1}^{K} C_{ji}^{\beta\phi} \cdot a_i^\phi$. Similarly, the expression $\mathbf{J}_{ji}((a_j)^l, (a_i)^l)$ becomes $\sum_{\kappa,\phi}^{K} C_{ji}^{\kappa\phi} \cdot a_j^\kappa \cdot a_i^\phi$.



($*^5$) : According to Eq.(1.2), only one element of the sum $\sum_{\kappa}^{K} (a_j^\kappa)^l$ is not zero and has a value of 1. This gives the following equivalence:

$$\log\left[\exp\left(\sum_{\kappa}^{K}(a_j^{\kappa})^l \cdot F(\kappa)\right)\right] = \log\left[\exp\left(1 \cdot F(\kappa)\right)\right] = 1 \cdot \log\left[\exp\left(F(\kappa)\right)\right]$$
$$= \sum_{\kappa}^{K}(a_j^{\kappa})^l \log\left[\exp\left(F(\kappa)\right)\right]$$

with $F(\kappa)$ be any expression of κ

Appendix B

Ising Gauge details

$(*^6)$

$$\begin{aligned}
\sum_{\kappa^\star}^{K_j} W_{ji}^{(1')}(\kappa^\star,\phi^\star) &= \sum_{\kappa^\star}^{K_j} W_{ji}^{(1)}(\kappa^\star,\phi^\star) - \frac{1}{K_j}\sum_{\kappa^\star}^{K_j}\sum_{\kappa}^{K_j} W_{ji}^{(1)}(\kappa,\phi^\star) \\
&\quad - \frac{1}{K_i}\sum_{\kappa^\star}^{K_j}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa^\star,\phi) + \frac{1}{K_iK_j}\sum_{\kappa^\star}^{K_j}\left(\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa,\phi)\right) \\
&= \sum_{\kappa^\star}^{K_j} W_{ji}^{(1)}(\kappa^\star,\phi^\star) - \frac{K_j}{K_j}\sum_{\kappa}^{K} W_{ji}^{(1)}(\kappa,\phi^\star) \\
&\quad - \frac{1}{K_i}\sum_{\kappa^\star}^{K_j}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa^\star,\phi) + \frac{K_j}{K_iK_j}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa,\phi) \\
&= 0
\end{aligned}$$

$(*^7)$

$$\begin{aligned}
\sum_{\phi^\star}^{K_i} W_{ji}^{(1')}(\kappa^\star,\phi^\star) &= \sum_{\phi^\star}^{K_i} W_{ji}^{(1)}(\kappa^\star,\phi^\star) - \frac{1}{K_j}\sum_{\phi^\star}^{K_i}\sum_{\kappa}^{K_j} W_{ji}^{(1)}(\kappa,\phi^\star) \\
&\quad - \frac{1}{K_i}\sum_{\phi^\star}^{K_i}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa^\star,\phi) + \frac{1}{K_iK_j}\sum_{\phi^\star}^{K_i}\left(\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa,\phi)\right) \\
&= \sum_{\phi^\star}^{K_j} W_{ji}^{(1)}(\kappa^\star,\phi^\star) - \frac{1}{K_j}\sum_{\phi^\star}^{K_i}\sum_{\kappa}^{K_j} W_{ji}^{(1)}(\kappa,\phi^\star) \\
&\quad - \frac{K_i}{K_i}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa^\star,\phi) + \frac{K_i}{K_iK_j}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W_{ji}^{(1)}(\kappa,\phi) \\
&= 0
\end{aligned}$$

$*^8$

$$\begin{aligned}\sum_{\kappa^\star}^{K_j} W^{(2')}_{jiy}(\kappa^\star,\phi^\star,\gamma^\star) &= \sum_{\kappa^\star}^{K_j} W^{(2)}_{jiy}(\kappa^\star,\phi^\star,\gamma^\star) - \frac{K_j}{K_j}\sum_{\kappa}^{K_j} W^{(2)}_{jiy}(\kappa,\phi^\star,\gamma^\star)\\ &- \frac{1}{K_i}\sum_{\kappa^\star}^{K_j}\sum_{\phi}^{K_i} W^{(2)}_{jiy}(\kappa^\star,\phi,\gamma^\star) - \frac{1}{K_y}\sum_{\kappa^\star}^{K_j}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa^\star,\phi^\star,\gamma)\\ &+ \frac{K_j}{K_jK_i}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W^{(2)}_{jiy}(\kappa,\phi,\gamma^\star) + \frac{1}{K_iK_y}\sum_{\kappa^\star}^{K_j}\sum_{\phi}^{K_i}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa^\star,\phi,\gamma)\\ &+ \frac{K_j}{K_yK_j}\sum_{\gamma}^{K_y}\sum_{\kappa}^{K_j} W^{(2)}_{jiy}(\kappa,\phi^\star,\gamma) - \frac{K_j}{K_jK_iK_y}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa,\phi,\gamma)\\ &= 0\end{aligned}$$

$*^9$

$$\begin{aligned}\sum_{\phi^\star}^{K_i} W^{(2')}_{jiy}(\kappa^\star,\phi^\star,\gamma^\star) &= \sum_{\phi^\star}^{K_i} W^{(2)}_{jiy}(\kappa^\star,\phi^\star,\gamma^\star) - \frac{1}{K_j}\sum_{\phi^\star}^{K_i}\sum_{\kappa}^{K_j} W^{(2)}_{jiy}(\kappa,\phi^\star,\gamma^\star)\\ &- \frac{K_i}{K_i}\sum_{\phi}^{K_i} W^{(2)}_{jiy}(\kappa^\star,\phi,\gamma^\star) - \frac{1}{K_y}\sum_{\phi^\star}^{K_i}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa^\star,\phi^\star,\gamma)\\ &+ \frac{K_i}{K_jK_i}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W^{(2)}_{jiy}(\kappa,\phi,\gamma^\star) + \frac{K_i}{K_iK_y}\sum_{\phi}^{K_i}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa^\star,\phi,\gamma)\\ &+ \frac{1}{K_yK_j}\sum_{\phi^\star}^{K_i}\sum_{\gamma}^{K_y}\sum_{\kappa}^{K_j} W^{(2)}_{jiy}(\kappa,\phi^\star,\gamma) - \frac{K_i}{K_jK_iK_y}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa,\phi,\gamma)\\ &= 0\end{aligned}$$

$*^{10}$

$$\begin{aligned}\sum_{\gamma^\star}^{K_y} W^{(2')}_{jiy}(\kappa^\star,\phi^\star,\gamma^\star) &= \sum_{\gamma^\star}^{K_y} W^{(2)}_{jiy}(\kappa^\star,\phi^\star,\gamma^\star) - \frac{1}{K_j}\sum_{\gamma^\star}^{K_y}\sum_{\kappa}^{K_j} W^{(2)}_{jiy}(\kappa,\phi^\star,\gamma^\star)\\ &- \frac{1}{K_i}\sum_{\gamma^\star}^{K_y}\sum_{\phi}^{K_i} W^{(2)}_{jiy}(\kappa^\star,\phi,\gamma^\star) - \frac{K_y}{K_y}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa^\star,\phi^\star,\gamma)\\ &+ \frac{1}{K_jK_i}\sum_{\gamma^\star}^{K_y}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i} W^{(2)}_{jiy}(\kappa,\phi,\gamma^\star) + \frac{K_y}{K_iK_y}\sum_{\phi}^{K_i}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa^\star,\phi,\gamma)\\ &+ \frac{K_y}{K_yK_j}\sum_{\gamma}^{K_y}\sum_{\kappa}^{K_j} W^{(2)}_{jiy}(\kappa,\phi^\star,\gamma) - \frac{K_y}{K_jK_iK_y}\sum_{\kappa}^{K_j}\sum_{\phi}^{K_i}\sum_{\gamma}^{K_y} W^{(2)}_{jiy}(\kappa,\phi,\gamma)\\ &= 0\end{aligned}$$

Appendix C

Absolute errors details

$*^{11}$
The $K \times K$ weights after ising $W_{ji}^{(1')}(\kappa^\star, \phi^\star)$ from Eq.(2.17) can be written as:

$$
\begin{aligned}
W_{ji}^{(1')}(\kappa^\star, \phi^\star) = W_{ji}^{(1)}(\kappa^\star, \phi^\star) - \frac{1}{K_j} W_{ji}^{(1)}(\kappa^\star, \phi^\star)) - \frac{1}{K_j} \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} W_{ji}^{(1)}(\kappa, \phi^\star)) \\
- \frac{1}{K_i} W_{ji}^{(1)}(\kappa^\star, \phi^\star) - \frac{1}{K_i} \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_i} W_{ji}^{(1)}(\kappa^\star, \phi) \\
+ \frac{1}{K_j K_i} \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} W_{ji}^{(1)}(\kappa, \phi^\star) + \frac{1}{K_j K_i} \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_i} W_{ji}^{(1)}(\kappa^\star, \phi) \\
+ \frac{1}{K_j K_i} W_{ji}^{(1)}(\kappa^\star, \phi^\star) + \frac{1}{K_j K_i} \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_i} W_{ji}^{(1)}(\kappa, \phi) \\
= (1 + \frac{1}{K_j K_i} - \frac{1}{K_j} - \frac{1}{K_i}) W_{ji}^{(1)}(\kappa^\star, \phi^\star) \\
+ (-\frac{1}{K_j} + \frac{1}{K_j K_i}) \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} W_{ji}^{(1)}(\kappa, \phi^\star)) \\
+ (-\frac{1}{K_i} + \frac{1}{K_j K_i}) \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_i} W_{ji}^{(1)}(\kappa^\star, \phi) + \frac{1}{K_j K_i} \sum_{\substack{\kappa \\ \kappa \neq \kappa^\star}}^{K_j} \sum_{\substack{\phi \\ \phi \neq \phi^\star}}^{K_i} W_{ji}^{(1)}(\kappa, \phi)
\end{aligned}
$$

$*^{12}$

$$\begin{cases} \dfrac{\partial C_{j,i}}{\partial F_{ji}} = 1 - \dfrac{2\left(F_{ji} + \sum_{\substack{s \\ s \neq i}}^{M} F_{js}\right) \sum_{r,s} F_{r,s} - (\sum_s F_{j,s})^2}{(\sum_{r,s} F_{r,s})^2} & \\ \dfrac{\partial C_{ji}}{\partial F_{js}} = \dfrac{2\left(F_{ji} + \sum_{\substack{s \\ s \neq i}}^{M} F_{js}\right) \sum_{r,s} F_{r,s} - (\sum_s F_{j,s})^2 \cdot 2}{(\sum_{r,s} F_{r,s})^2} & \forall s \in [1, M]/i \\ \dfrac{\partial C_{ji}}{\partial F_{rs}} = \dfrac{(\sum_s F_{j,s})^2}{(\sum_{r,s} F_{r,s})^2} & \forall r \in [1, M]/j \end{cases}$$

Bibliography

[1] Masato Ikeda Atsushi Yokota. *Amino Acid Fermentation*. Advances in Biochemical Engineering/Biotechnology. Springer Tokyo, 2017. ISBN 9784431565208. URL https://link.springer.com/book/10.1007/978-4-431-56520-8#bibliographic-information.

[2] Mourra-Díaz C. M. Farías-Rico, J. A. A short tale of the origin of proteins and ribosome evolution. *Microorganisms*, 10 2022. URL https://doi.org/10.3390/microorganisms10112115.

[3] Lynn Yarris. A new guide to exploring the protein universe. *science@berkley lab*, 3 2005. URL https://www2.lbl.gov/Science-Articles/Archive/sabl/2005/March/02-protein-universe.html.

[4] Leszek Konieczny and Irena Roterman. Chapter 3 - information encoded in protein structure. In Irena Roterman-Konieczna, editor, *From Globular Proteins to Amyloids*, pages 27–39. Elsevier, 2020. ISBN 978-0-08-102981-7. doi: https://doi.org/10.1016/B978-0-08-102981-7.00003-8. URL https://www.sciencedirect.com/science/article/pii/B9780081029817000038.

[5] Kaiyu Qiu, Nir Ben-Tal, and Rachel Kolodny. Similar protein segments shared between domains of different evolutionary lineages. *Protein Sci.*, 31(9):e4407, September 2022.

[6] Fatima Ardito, Michele Giuliani, Donatella Perrone, Giuseppe Troiano, and Lorenzo Lo Muzio. The crucial role of protein phosphorylation in cell signaling and its use as targeted therapy (review). *Int. J. Mol. Med.*, 40(2):271–280, August 2017.

[7] Personal figures.

[8] G.M. Cooper. *The Cell: A Molecular Approach*. The Cell: A Molecular Approach. ASM Press, 2000. ISBN 9780878931064. URL https://books.google.ch/books?id=DCdyQgAACAAJ.

[9] Mikhail Borisovich Evgen'ev. Heat shock proteins: a history of study in russia. *Cell Stress Chaperones*, 26(4), July 2021.

[10] M P Mayer and B Bukau. Hsp70 chaperones: cellular functions and molecular mechanism. *Cell. Mol. Life Sci.*, 62(6), March 2005.

[11] M B B Gutierres, C B C Bonorino, and M M Rigo. ChaperISM: improved chaperone binding prediction using position-independent scoring matrices. *Bioinformatics*, 36(3), February 2020.

[12] Micael Evgenev, D. Garbuz, and Olga Zatsepina. *Heat Shock Proteins and Whole Body Adaptation to Extreme Environments*. 01 2014. ISBN 978-94-017-9234-9. doi: 10.1007/978-94-017-9235-6.

[13] Matthias P. Mayer and Lila M. Gierasch. Recent advances in the structural and mechanistic aspects of hsp70 molecular chaperones. *Journal of Biological Chemistry*, 294(6):2085–2097, 2019. ISSN 0021-9258. doi: https://doi.org/10.1074/jbc.REV118.002810. URL https://www.sciencedirect.com/science/article/pii/S0021925820368538.

[14] K M Flaherty, C DeLuca-Flaherty, and D B McKay. Three-dimensional structure of the ATPase fragment of a 70K heat-shock cognate protein. *Nature*, 346(6285), August 1990.

[15] Jose Carlos Solana, Lorena Bernardo, Javier Moreno, Begoña Aguado, and Jose M Requena. The astonishing large family of HSP40/DnaJ proteins existing in leishmania. *Genes (Basel)*, 13(5), April 2022.

[16] Ciamak Ghazaei. Role and mechanism of the hsp70 molecular chaperone machines in bacterial pathogens. *J. Med. Microbiol.*, 66(3):259–265, March 2017.

[17] Yan Huo, Zhiyu Song, Haiting Wang, Ziyu Zhang, Na Xiao, Rongxiang Fang, Yuman Zhang, and Lili Zhang. GrpE is involved in mitochondrial function and is an effective target for RNAi-mediated pest and arbovirus control. *Insect Mol. Biol.*, 31(3):377–390, June 2022.

[18] Julia Behnke, Matthias J Feige, and Linda M Hendershot. BiP and its nucleotide exchange factors grp170 and sil1: mechanisms of action and biological functions. *J. Mol. Biol.*, 427(7):1589–1608, April 2015.

[19] Melissa J. Mann, Chris Melendez-Suchi, Maria Sukhoplyasova, Ashley R. Flory, Mary Carson Irvine, Anuradha R. Iyer, Hannah Vorndran, Christopher J. Guerriero, Jeffrey L. Brodsky, Linda M. Hendershot, and Teresa M. Buck. Loss of grp170 results in catastrophic disruption of endoplasmic reticulum functions. *bioRxiv*, 2023. doi: 10.1101/2023.10.19.563191. URL https://www.biorxiv.org/content/early/2023/10/20/2023.10.19.563191.

[20] André Zapun, Claude A Jakob, David Y Thomas, and John JM Bergeron. Protein folding in a specialized compartment: the endoplasmic reticulum. *Structure*, 7(8):R173–R182, 1999. ISSN 0969-2126. doi: https://doi.org/10.1016/S0969-2126(99)80112-9. URL https://www.sciencedirect.com/science/article/pii/S0969212699801129.

[21] Nitu L. Wankhede, Mayur B. Kale, Aman B. Upaganlawar, Brijesh G. Taksande, Milind J. Umekar, Tapan Behl, Ahmed A.H. Abdellatif, Prasanna Mohana Bhaskaran, Sudarshan Reddy Dachani, Aayush Sehgal, Sukhbir Singh, Neelam Sharma, Hafiz A. Makeen, Mohammed Albratty, Hamed Ghaleb Dailah, Saurabh Bhatia, Ahmed Al-Harrasi, and Simona Bungau. Involvement of molecular chaperone in protein-misfolding brain diseases. *Biomedicine Pharmacotherapy*, 147:112647, 2022. ISSN 0753-3322. doi: https://doi.org/10.1016/j.biopha.2022.112647. URL https://www.sciencedirect.com/science/article/pii/S075333222200035X.

[22] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A A Kohl, Andrew J Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman,

Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021.

[23] Paolo De Los Rios Zoé Majeux. Exploring the sequence landscape of proteins with direct couplings analysis and pseudolikelihood maximisation. *Master of specialization in the Laboratory of Statistical Biophysics*, januar 2024. URL https://github.com/zoemaj/DCA.

[24] Magnus Ekeberg. Detecting contacts in protein folds by solving the inverse potts problem - a pseudolikelihood approach. Master's thesis, 2012.

[25] Anders Larsson. AliView: a fast and lightweight alignment viewer and editor for large datasets. *Bioinformatics*, 30(22):3276–3278, November 2014.

[26] Simona Cocco, Christoph Feinauer, Matteo Figliuzzi, Rémi Monasson, and Martin Weigt. Inverse statistical physics of protein sequences: a key issues review. *Reports on Progress in Physics*, 81(3):032601, January 2018. ISSN 1361-6633. doi: 10.1088/1361-6633/aa9965. URL http://dx.doi.org/10.1088/1361-6633/aa9965.

[27] Matteo Figliuzzi, Pierre Barrat-Charlaix, and Martin Weigt. How pairwise coevolutionary models capture the collective residue variability in proteins? *Mol. Biol. Evol.*, 35(4):1018–1027, April 2018.

[28] Jorge López Puga, Martin Krzywinski, and Naomi Altman. Bayes' theorem. *Nat. Methods*, 12(4):277–278, April 2015.

[29] Uniprot Consortium. *Nucleic Acids Res.*, 51(D1), January 2023.

[30] Hidden markov model based tool (hmmer). URL http://hmmer.org.

[31] Jaina Mistry, Sara Chuguransky, Lowri Williams, Matloob Qureshi, Gustavo A Salazar, Erik L L Sonnhammer, Silvio C E Tosatto, Lisanna Paladin, Shriya Raj, Lorna J Richardson, Robert D Finn, and Alex Bateman. Pfam: The protein families database in 2021. *Nucleic Acids Res.*, 49(D1):D412–D419, January 2021.

[32] Typhaine Paysan-Lafosse, Matthias Blum, Sara Chuguransky, Tiago Grego, Beatriz Lázaro Pinto, Gustavo A Salazar, Maxwell L Bileschi, Peer Bork, Alan Bridge, Lucy Colwell, Julian Gough, Daniel H Haft, Ivica Letunić, Aron Marchler-Bauer, Huaiyu Mi, Darren A Natale, Christine A Orengo, Arun P Pandurangan, Catherine Rivoire, Christian J A Sigrist, Ian Sillitoe, Narmada Thanki, Paul D Thomas, Silvio C E Tosatto, Cathy H Wu, and Alex Bateman. InterPro in 2022. *Nucleic Acids Res.*, 51(D1):D418–D427, January 2023.

[33] Jun Lu. Gradient descent, stochastic optimization, and other tales, 2024.

[34] Simona Cocco, Christoph Feinauer, Matteo Figliuzzi, Rémi Monasson, and Martin Weigt. Inverse statistical physics of protein sequences: a key issues review. *Rep. Prog. Phys.*, 81(3):032601, March 2018.

[35] Nicola Dietler, Umberto Lupo, and Anne-Florence Bitbol. Impact of phylogeny on structural contact inference from protein sequence data. *J. R. Soc. Interface*, 20(199):20220707, February 2023.

[36] Ucsf Chimerax, E C Meng, T D Goddard, E F Pettersen, G S Couch, Z J Pearson, J H Morris, and T E Ferrin. Tools for structure building and analysis. 32, 2023.

[37] Duccio Malinverni and Alessandro Barducci. *Coevolutionary Analysis of Protein Sequences for Molecular Modeling*, pages 379–397. Springer New York, New York, NY, 2019. ISBN 978-1-4939-9608-7. doi: 10.1007/978-1-4939-9608-7_16. URL https://doi.org/10.1007/978-1-4939-9608-7_16.

[38] Anne E Osbourn and Ben Field. Operons. *Cell. Mol. Life Sci.*, 66(23):3755–3775, December 2009.

[39] Hafumi Nishi, Alexey Shaytan, and Anna R Panchenko. Physicochemical mechanisms of protein regulation by phosphorylation. *Front. Genet.*, 5:270, August 2014.