Alma Mater Studiorum – Università di Bologna

FACOLTÀ DI SCIENZE MATEMATICHE FISICHE E NATURALI
DOTTORATO DI RICERCA IN MATEMATICA, XX CICLO

MAT 07: Fisica Matematica

Complexity Measures and Similarity Metrics: Properties and Applications to Biological Signals

Chiara Farinelli

Coordinatore:
Alberto Parmeggiani

Relatore:
Mirko Degli Esposti

Esame Finale anno 2008

A Luigi

"There are more things in heaven and eart, Horatio,
than are dreamt of your phylosophy"...

William Shakespeare (Hamlet, scene V)

Introduction

Literary texts, images, biological signals such as DNA or protein sequences, ECG (Electrocardiogram)... there are many things in real life that can be represented by symbolic sequences.

Some sequences hide more information in them beyond the single characters that make them up: a Shakespeare text, for example, can be stored in a long sequence of twenty-six or more alphabetic characters (including punctuation, spaces and so on), but the semantic sense is not directly gained from the simple character by character reading. Each character forms words that all together create the phrases.

Similarly for biological sequences: the alphabet of proteins is formed by the twenty amino acids that, according to hidden rules, bind together to generate the phrases (secondary structures) founding the tertiary structure (and hence its meaning, i.e. its functionality).

In particular, in the field of biological signals, where the information necessary to build the final objects seems hidden in the sequence itself, the analysis of the sequence through mathematical instruments able to capture the building rules of the strings (or "complexity" of the strings) seems very promising. The use of distances or similarity measures, can be seen as another useful instrument for extracting some kind of information from signals.

Which kind of information is captured depends, clearly, by the signals under observation and by the task that is fixed from time to time.

In case of biological signals, these procedures may have a meaningful exploitation. Just to name a few cases, if we consider cardiological signals, the relevance of having methods that are able to make a clustering of patients according to some pathology is evident. While, for protein sequences it can be useful the automatic classification of the proteins according to their structural classes.

Each of the two worlds, mathematics and biology, interacting can provide ideas, questions and new possible research directions to each other. The biology request for tools, for istance, that are able to analyze the huge amount of data available nowdays, or the need to have algorithms of *attribution, classification* and *clustering*, leads to develop new mathematical methods or, sometimes, to readjust methods/theorems coming out of different fields.

On the other hand, a deeper comprehension of the methodologies used in biological signals analysis may take us to a better comprehension of the achieved results and, finally, to a greater contribution to the solution of the starting biological problem.

By the way: "Living things are too beautiful for there not to be a mathematics that describes them"! (Tom Schneider)

In this thesis both the study of the mathematical properties of some *complexity indicators* and of *similarity metrics* are faced, with main focus on the applications to biological signals.

A series of results coming from the Information Theory, Dynamical Systems, and from the Statistical field have been re-elaborated, studied thoroughly and compared from the mathematical point of view.

For what concerns the complexity measures, this thesis is principally focused on the *parsing* of strings, a particular kind of rule for cutting the string in substrings.

Little adaption of these parsings, combined with suitable coding of the words, leads to the very well known compression programs used daily for

zipping files on the computer.

In **Chapter 1** we analyze in detail the *exhaustive* parsing (or *LZ complexity*) of a string S following the approach of Lempel and Ziv in [72] that examines the concept of complexity as related to the rate of vocabulary growth. The relations between the exhaustive parsing and more known parsings (LZ77, LZ78) are investigated, and it is shown the optimality of the exhaustive parsing for ergodic sources.

In **Chapter 2** we analyze similarity metrics related to different teoretical approaches. The first part of the chapter concerns the so called "Information Distances", based on the Kolomogorov complexity: the main features and properties, following the Vitany approach [77], are recalled, togheter with the approximation of this uncomputable distance using the compressor algorithms, namely the Normalized Compression Distance (NCD) or Universal Similarity Metric (USM) [76, 24].

The LZ-distance [90], mainly used in our applications, is shown in this section and it is very similar to the NCD distance: instead of the compression of a string, the quantity involved in this distance is the number of components obtained with the exhaustive parsing on a string.

The second part of this chapter analyzes the distances related to the Relative Entropy (or Kullback-Leibler divergence), an indicator of the discrepancy between two different sources. We face the pratical problem of estimating it showing different approximation approaches, based on parsing [115], compression procedure [3] or the BWT transformation [9].

Finally it is shown a distance (the n-gram distance), based on the n-gram frequencies of two sequences that, togheter with the LZ distance, will be used in our applications.

The last two chapters are devoted to the biological sequences.

In **Chapter 3** the methods previously shown are applied on cardiological signals. The main results of this chapter are taken from two pubblications of us ([27, 28]).

On this kind of sequences we used the exhaustive parsing for exploring if

the LZ complexity of a cardiological signal is able to capture some features owned by a signal and not by another with the goal of classifying/clustering patients according different pathologies.

Having seen a real discriminatory capability connected to this indicator, we moved on using the LZ distance based on such exhaustive parsing where an improvement in the attributions has been noticed.

We are also focused on wheter the choice of coding the ECG signal in the bynary HRV is really optimal, exploring different possible codes.

Finally, in the last section of the chapter are shown new results achieved on the same data with the use of the n-gram distance.

For what concerns the protein sequences, this is a very recent field in which we are moving. Primary protein sequences are without a doubt a very intersting kind of sequences in which are probabily hidden useful information for the folding of the protein itself.

Several computational methods that try to classify proteins according to their structural class (and finally to give some answers to the folding problem) have been developed in the last years.

In **Chapter 4**, after the introduction of some biological concepts, we show some of these alignment free methods, underling mathematical equivalences/differences between some methods and trying to compare the results.

The comparison is done using the results given in the literature with the purpose of understanding the state of the art in structural class prediction methods and find out the promising methods and codes of the primary sequence that are able to capture some protein's features playing a bigger role in the protein folding.

The analysis done in this chapter has shown that the comparison of the results is very hard: different methods are tested on different datasets, some datasets are statistically poorly significant, different procedures are used for measuring the prediction (classification) quality of sequences into structural classes, and so on.

However, a comparison of the results is the main way we have for test-

ing the methods or distances: contingent mathematical equalities (proved on markov sequences or assumption of gaussian distributions), do not automatically imply an equivalence in the real experimental results. The protein sequences, indeed, are not markov sequences nor variables with gaussian distribution, so in the real case the *try and see* approach is the real test.

Contents

Chapter 1
Parsing of a symbolic sequence

Parsing techinques are directly connected to creation of a vocabulary of recurrent subwords within the sequence.
The information stored in these parsings can have several and related applications, such as compression, quantitative estimate of (absolute) complexity and, most important for us, implementation of heuristic conditional entropy/information estimators or similarity (pseudo-)distances between different strings or different classes/sources of strings.
In this section we analyze in detail the *exhaustive* parsing, following the approach of Lempel and Ziv in [72] that examines the concept of complexity as related to the rate of vocabulary growth.
With the auxilio of the definition of eigenvalues and eigenvocabularies, the relations between the exhaustive parsing and more known parsings (lz77, lz78) are investigated.
Finally, the optimality of the exhaustive parsing for ergodic sources is shown.

1.1 Production and Reproduction processes

We start by fixing some notations. $\mathcal{A}$ will denote a finite alphabet, $\mathcal{A}_n$ all possible words of length n and we let $\mathcal{A}^* = \bigcup_{n\in\mathbb{N}} \mathcal{A}_n$. For some of the applications and for all the mathematical results we will assume w.l.g. $\mathcal{A} = \{0, 1\}$, but sometimes it will be useful to consider the ASCII alphabet or the DNA nucleotides set $\mathcal{A} = \{A, C, G, T\}$. For any $x \in \mathcal{A}_n$, we denote by $Dic(x)$ all distinct substrings of x.

Given any finite string $x = x_1 x_2 \cdots x_n$, a **parsing** $P(x)$ of x is any given ordered partition of x into small words: $0 \cdot 10 \cdot 111$ and $0 \cdot 1 \cdot 0 \cdot 11 \cdot 1$ are two different parsing of the string $x = 010111$. A parsing into different words is called a *distinct parsing.*

In the following, we shall refer to an (information) source as to any dynamical system $(\mathcal{A}^{\mathbb{N}}, T, \mu)$, where T is the usual shift on the space of infinite sequences $\mathcal{A}^{\mathbb{N}}$ with probability measure μ and entropy $h = h(\mu)$.

The methods we are going to review are not directly based on the global statistics of words, but they focus on the string as a direct result of an ordered sequence of elementary production processes generating the string.

Definition 1.1.1. *Given a sequence x of length n and any $i < j \leqslant n$, let us denote the substring $x_i \dots x_{j-1} x_j$ by $x(i, j)$; a sequence x is (strictly)* **producible** *by its proper prefix $x(1, j)$, if x can be obtained by concatenating $x(1, j)$ with the results of a sequential copying process starting at position k, $k < j - 1$, plus an additional character of new information at the end of the string. We denote this process by $x(1, j) \Rightarrow x$.*

A sequence x of length n is **reproducible** *by its proper prefix $x(1, j)$, when exists a position k in $x(1, j)$ such that $x(k, l(x(j + 1, n) + k - 1) = x(j + 1, n)$ (where $l(x(j + 1, n)$ is the length of the suffix of x). We denote this process by $x(1, j) \longrightarrow x$,*

For instance, if $x = 01001001001$, then $x(1, 5)$ *reproduces* x with pointer $k = 3$: $x(1, 5) \longrightarrow x$. Instead, if $x = 01001001000$, then $x(1, 5)$ *produces* x with pointer $k = 3$: $x(1, 5) \Rightarrow x$.

Remark 1.1.1. *From the definition* (1.1.1)*, producibility implies reproducibility. Infact, if a string $x = x(1, n)$ is producibile by its proper prefix $x(1, j)$ than the string $x(1, n - 1)$ is reproducibile by its proper prefix. The contrary implication is not true: Reproducibility does'nt imply the producibility.*

According to the above example, if $x = 01001001001$ then
$x(1, 5) \longrightarrow x(1, n - 1) = 0100100100$

Definition 1.1.2. *A history $\mathcal{H}(x)$ of x is a parsing of the string which describes a sequence of legal production processes: $\mathcal{H}(x) = x(1, j_1)x(j_1 + 1, j_2)\cdots x(j_k + 1, n)$, such that $x(1, j_m) \Rightarrow x(1, j_{m+1})$, that is:*

$$\exists\ Q \in Dic(x(1, h_{i+1} - 2))\ s.t.\ x(1, h_{i+1} - 1) = x(1, h_i)Q\ .$$

We denote by $c_{\mathcal{H}}(x)$ the number of phrases in a history $\mathcal{H}(x)$

1.2 The exhaustive parsing

Definition 1.2.1. *[72] The LZ- complexity of the string x is the least possible number of steps in which x can be generated according to the rule of a production process:*

$$c(x) = \min_{\mathcal{H}\ history\ of\ x} c_{\mathcal{H}}(x)$$

Definition 1.2.2. *A production step from $x(1, h_{i-1})$ to $x(1, h_i)$ is* **exhaustive** *if $x(1, h_i)$ is producible but not reproducible from $x(1, h_{i-1})$. We denote by $E(x)$ the exhaustive parsing of x, that is the history $\mathcal{H}(x) = x(1, j_1)x(j_1 + 1, j_2)\cdots x(j_k+1, n)$ where each step is exhaustive, with the possible exception of the last one.*

In the following theorem we resume the principal properties of this parsing

Theorem 1.2.1. *(i) Every nonnull sequence x has a unique exhaustive history*

(ii) For each $x \in \mathcal{A}^*$, $c(x) = c_E(x)$, where $E(x)$ is the unique exhaustive history of x

(iii) (Subadditivity of exhaustive parsing) $\forall x, y \in \mathcal{A}^*$, xy denotes the concatenation of the two strings and:

$$c(xy) \leqslant c(x) + c(y)$$

(*iv*) *For any* $x \in \mathcal{A}_n$,

$$c(x) < \frac{n}{(1-\varepsilon_n)\log n}, \quad \varepsilon_n = 2\frac{1+\log\log(n|\mathcal{A}|)}{\log n}$$

Proof. (*i*) By definition, an exhaustive production process always exists. Suppose $\mathcal{H}(x)$ and $\mathcal{K}(x)$ are two distinct exhaustive parsings of sequence x. Then, there is an index i such that

$$\begin{cases} x(1,h_j) = x(1,k_j) & \forall j < i \quad (i) \\ x(1,h_i) \subset x(1,k_i) \text{ AND } x(1,h_i) \neq x(1,k_i) & (ii) \end{cases} \tag{1}$$

(The second assumption may be taken without loss of generality)

By assumption, the two parsings $\mathcal{H}(x)$ and $\mathcal{K}(x)$ are exhaustive, then at step i we have:

- $x(1,h_i)$ is producible but not reproducible from $x(1,h_{i-1})$

and similarly

- $x(1,k_i)$ is producible but not reproducible from $x(1,k_{i-1})$.

Since $i-1 < i$, from property (*i*) of the theorem 1 we know that $x(1,h_{i-1}) = x(1,k_{i-1})$. Therefore, $x(1,h_i)$ is not reproducible from $x(1,k_{i-1})$.

But by property (*ii*) of the theorem 1, we have that $x(1,h_i) \subset x(1,k_i)$ and $x(1,h_i) \neq x(1,k_i)$, from which we have that

$$x(1,h_i) \subseteq x(1,k_i-1)\ . \tag{2}$$

Finally, notice that the assumtpion of producibility of $x(1,k_i)$ from $x(1,k_{i-1})$ means that $x(1,k_{i-1})$ is reproducible from $x(1,k_i-1)$. The latter is in contrast with (2):

- if $x(1,h_i) = x(1,k_i-1)$ the absurd is trivial
- if $x(1,h_i) \neq x(1,k_i-1)$, since up to h_{i-1} the two parsings coincide, then $x(1,h_i) \subset x(1,k_i-1)$ and it is not possible that $x(1,h_i)$ is not reproducible from $x(1,k_i-1)$.

Thus, the exhaustive parsing is unique.

(*ii*) (This is Theorem 1 of [72]).

(*iii*) (This is Theorem 4 of [72]) Let $E(x)$ and $E(y)$ be the exhaustive histories of x and y respectively.

Now, we consider the concatenation of the two histories $E(x) \wedge E(y)$ that is a history $\mathcal{H}(xy)$ of the string $x \wedge y$. Infact

$$\mathcal{H}(xy) = x(1, j_1)x(j_1{+}1, j_2) \dots x(j_k{+}1, n)y(1, i_1)y(i_1{+}1, i_2)1 \dots y(i_k{+}1, n)$$

It is easy to see that all the components in x are such that $x(1, j_m) \Rightarrow x(1, j_{m+1})$ (because they are components of the exhaustive history $E(x)$). When we begin to read the string y, as well, all (or none of) the characters in the component $y(1, i_m)$ may have already appeared in x. In the second case (all the characters in $y(1, i_m)$ haven't already appeared in x) the components of the history $H(xy)$ are equal to the components of the exhaustive component of the history $H(x) + H(y)$; while, in the first case, for constructing the exhaustive component in xy we must go on reading y until we find a new character that has not previously appeared. The component so formed is longer than the component $y(1, i_m)$ from which we started.

Then, it is clear that the *number* of components obtained with the exhaustive parsing on xy is smaller or equal to the number of components obtained summing those from x and y separately.

This implies:

$$c(xy) = c_E(xy) \leqslant c_{\mathcal{H}}(xy) = c_E(x) + c_E(y) = c(x) + c(y)$$

(*iv*) This is Theorem 2 of [72].

□

1.3 Eigenvocabulary, Eigenvalue, Eigensequence

We now want to quantify the *production of new information* along the sequence: a word of a vocabulary $e \in \mathrm{Dic}(x)$ is called an *eigenword* if it is *not* contained in the vocabulary of any proper prefix of x. More formally, following [72], we denote by $x\pi^j = x(1, n-j)$, where n is the length of x and $0 \leqslant j \leqslant n$, $x\pi^0 = x$ and $x\pi^n$ is the empty word. Then, $\mathrm{Dic}(x\pi^j) \subseteq \mathrm{Dic}(x\pi^k)$ for all $0 \leqslant k \leqslant j \leqslant n$ and $e \in \mathrm{Dic}(x)$ is an eigenword if and only if $e \in \mathrm{Dic}(x) \setminus \mathrm{Dic}(x\pi)$.

Using these notations, it is possible to redefine the production and reproduction processes [72]:

Definition 1.3.1. *If $j < n$, then a prefix $x(1, j)$ produces $x = x(1, n)$ if there is a word $w \in Dic(x\pi^2)$ such that $x\pi = x(1, j)w$, that is $w = x(j + 1, n - 1) \in Dic(x\pi^2)$. Since the producibility is a generalization of the so-called reproducibility, a sequence z is reproducible from its proper prefix y if there is a sequence w such that $z = yw$ and $w \in Dic(z\pi)$.*

Definition 1.3.2. *The set $\lambda Dic(x) = Dic(x) \setminus Dic(x\pi)$ is called eigenvocabulary and its cardinality $1 \leqslant \lambda(x) \leqslant n$ is the eigenvalue of the string $x \in \mathcal{A}_n$.*
The eigensequence $(\lambda_x(i))_{i=1,\ldots,n}$ of $x(1, n)$ is the sequence of the eigenvalues at each site $i = 1, \ldots, n$ in $x(1, n)$: $\lambda_x(i) \doteq \lambda(x(1, i))$.

For example, for the string $x = 0010$ we obtain:

$Dic(x) = \{0, 1, 00, 01, 10, 001, 010, 0010\}$

$Dic(x\pi) = \{0, 1, 00, 01, 001\}$

$\lambda Dic(x) = \{10, 010, 0010\}$

$\lambda(x) = 3$

$\lambda_x(i) = (1, 1, 3, 3)$

Given the eigenvalue of x, the *eigenwords* (elements in $\lambda\mathrm{Dic}(x)$) are identified as follows (Theorem 6 in [72]):

$$e(x) \;=\; \{x(j,n)|\, 1 \leqslant j \leqslant \lambda(x)\},$$

Moreover, it is easy to see that (Lemma 4 in [72]):

$$\lambda(x\pi) \leqslant \lambda(x)$$

The growth rate of the eigenvocabulary indicates the effective new words produced along the string and it will be used as a natural index either of the *complexity* of the single string or (asymptotically when $n \to \infty$) of the *entropy* of the (ergodic) source emitting x.

Remark 1.3.1. *In an* **innovative** *history each step produces a larger eigenvocabulary w.r.t. the previous step, i.e. the eigensequence* $(\lambda_x(h_i))_{1\leqslant i\leqslant m}$ *of the eigenvalues at the parsing cut sites is strictly increasing.*

Definition 1.3.3. *A history component* $x(h_{i-1}+1, h_i)$ *is* **primitive** *if* h_i *is the least integer s.t. the eigenvalue of* $x(1, h_i)$ *is greater than that of* $x(1, h_{i-1})$*. We denote by prim*(x) *the primitive parsing of* x*, where each of its components, with the possible exception of the last one, is primitive.*

The following theorem provides a useful alternative definition of the above histories, using the characterization with the eigenvalues. The proof is in ref. [72].

Theorem 1.3.1 (Theorem 8, [72])**.** *Let* $\mathcal{H}(x) = x(1, h_1)x(h_1+1, h_2)\ldots x(h_{m-1}+1, h_m)$ *be some history of* x*.*

(*i*) $\mathcal{H}(x)$ *is primitive if and only if, for* $i = 1, 2, \ldots, m-1$

$$h_i = \min\{h \in M_x(i-1)\}$$

where

$$M_x(i-1) = \{j \mid j > h_{i-1}\ and \lambda_x(j) > \lambda_x(h_{i-1})\}\ .$$

(ii) $\mathcal{H}(x)$ *is exhaustive if and only if, for* $i = 1, 2, \ldots, m-1$

$$h_i = \min\{h \in N_x(i-1)\}$$

where

$$N_x(i-1) = \{j \mid j > h_{i-1} \text{ and } \lambda_x(j) > h_{i-1}\} \ .$$

Notice that, as in the standard definitions, the last history step may be or not be either primitive or exhaustive. □

With this notation we can now prove the point (ii) of theorem1.2.1 [72]:

$$c(x) = min_{\mathcal{H}} c_{\mathcal{H}}(x) = c_E(x)$$

Proof. Let $\mathcal{H}_x = \{h_1, h_2, \ldots, h_m\}$ with $c_{\mathcal{H}}(x) = m$ and $E_x = \{e_1, e_2, \ldots, e_k\}$ with $c_E(x) = k$ and $1 = h_1 < h_2 z \ldots < h_m = \ell(x)$, $1 = e_1, e_2, \ldots e_k = l(x)$.

Let η be a mapping from E_x into $\mathcal{H}_x$, defined by:

$$\eta(e_i) = \max\{h \in \mathcal{H}_x | h \leqslant e_i\} \ i = 1, 2, \ldots, k$$

We have that $\eta(e_1) = h_1$ and $\eta(e_k) = h_m = l(x)$. When $k > 2$ consider any i such that $2 \leqslant i \leqslant k-1$ and let $\eta(e_i) = h_j$. Then, it is clear that $j < m$ and, according to the definition, h_j is the bigger h such that $h_j \leqslant e_i$, $h_{j+1} > h_j$ must be such that $h_{j+1} > e_i$.

Then $e_{i-1} < h_j = \eta(e_i)$. Moreover, always from the definition of $\eta(e_i)$, we have that $\eta(e_i) = h_j \leqslant e_i$.

Hence, for each i such that $1 \leqslant i \leqslant k-1$ we have $e_{i-1} \leqslant \eta(e_i) \leqslant e_i$, therefore $\eta(e_i)$ is a one to one mapping from the set E_x onto a subset of $\mathcal{H}_x$ □

Remark 1.3.2. *Theorem 1.2.1 (ii) states that if* $f(x)$ *is the complexity of either LZ77 [114] or LZ78 [113] or primitive parsing and* $c(x)$ *is the LZ-complexity, then*

$$c(x) \leqslant f(x) \ .$$

1.4 Eigenvalues of different parsings

In this section we present some properties of different parsings according to the notation introducted in the above section.

Exhaustive and primitive parsings are not the only parsings generated by legal production processes. Here and in the following we refer to LZ77 algorithm in the so-called infinite-window version. It is a self-parsing procedure of a sequence into $m(x)$ distinct phrases such that each phrase is the shortest string which is not appeared in the past.

Proposition 1.4.1. *The parsings induced by LZ77 [114] and LZ78 [113] are histories of input sequence x.*

Proof. Let us denote any history component $x(h_{i-1}, h_i)$ of an m-step parsing by w_i, the i-th word in the parsing, for $i = 1, \ldots, m$ and $h_0 = 1$. Therefore, $x(1, h_{i+1}) = x(1, h_i)w_{i+1}$ and to prove that the parsing is a history, we have to prove that $w_{i+1}\pi \in Dic(x(1, h_{i+1} - 2))$, where $w_{i+1}\pi$ denotes the word w_{i+1} without its last symbol.

1. A new word in LZ77 parsing is w_{i+1} s.t. $w_{i+1}\pi \in Dic(x(1, h_i))$. With the possible exception of the alphabet words and of the last one, we can say that $Dic(x(1, h_i)) \subset Dic(x(1, h_{i+1} - 2))$, then this parsing is a history.

2. A new word in LZ78 parsing is w_{i+1} s.t. $w_{i+1}\pi \in \{w_1, \ldots, w_i\}$, then again $w_{i+1}\pi \in Dic(x(1, h_{i+1} - 2))$ and the parsing is a history.

□

Example 1.4.1. *In the following tables, different parsings and complexities are shown for two binary sequences S_1 and S_2 of length 20. Sequence S_1 is periodic, whereas sequence S_2 is patternless.*

We considered the exhaustive parsing, the LZ77 parsing, the LZ78 parsing and the primitive parsing. We recall that the complexity is defined as the length of the parsing.

$$S_1 = 01010101010101010101 = (01)^{10}$$

	PARSING	**COMPLEXITY**	
exhaustive	$0 \cdot 1 \cdot 010101010101010101$	LZ-complexity	3
LZ77	$0 \cdot 1 \cdot 010 \cdot 10101 \cdot 0101010101$	LZ77-complexity	5
LZ78	$0 \cdot 1 \cdot 01 \cdot 010 \cdot 10 \cdot 101 \cdot 0101 \cdot 0101$	LZ78-complexity	8
primitive	$0 \cdot 1 \cdot 010101010101010101$	prim-complexity	3

$$S_2 = 011110001101101110 10$$

	PARSING	**COMPLEXITY**	
exhaustive	$0 \cdot 1 \cdot 1110 \cdot 001 \cdot 101 \cdot 10111 \cdot 010$	LZ-complexity	7
LZ77	$0 \cdot 1 \cdot 11 \cdot 10 \cdot 00 \cdot 1101 \cdot 1011 \cdot 1010$	LZ77-complexity	8
LZ78	$0 \cdot 1 \cdot 11 \cdot 10 \cdot 00 \cdot 110 \cdot 1101 \cdot 11010$	LZ78-complexity	8
primitive	$0 \cdot 1 \cdot 1110 \cdot 0 \cdot 01 \cdot 10 \cdot 1 \cdot 10111 \cdot 01 \cdot 0$	prim-complexity	10

The following summarizes some properties of innovative, exhaustive and primitive parsings.

Theorem 1.4.1. *The following hold.*

(*i*) *Exhaustive and primitive parsings are innovative.*

(*ii*) *LZ77 and LZ78 parsings are not innovative.*

(*iii*) *For each* $x \in \mathcal{A}^*$

$$c_{prim}(x) = \max_{\substack{H \text{ innovative} \\ \text{history of } x}} c_H(x)$$

Proof. (*i*) As showed in Theorem 1.3.1 (*ii*), a production step from h_i to h_{i+1} is exhaustive if and only if $h_{i+1} \doteq \min\{h > h_i \mid \lambda_x(h) > h_i\}$, where

$\lambda_x(h)$ is the eigenvalue relative to $x(1,h)$. As previously seen in the definition of the eigenvalue of the string x of length n, $1 \leqslant \lambda(x) \leqslant n$. Similarly, if we take into account the sequence $x(1,h_i)$ of length h_i, we obtain that $1 \leqslant \lambda_x(h_i) \leqslant h_i$, than $\lambda_x(h_{i+1}) > h_i \geqslant l_x(h_i)$. Therefore the exhaustive history is innovative. Primitive parsing is innovative by definition.

(*ii*) Usually, the eigenvalues of two subsequent parsing steps (generated by either LZ77 or LZ78) are not strictly increasing. For instance, consider the two strings S_1 and S_2 defined above. In the following, we show the sequence of eigenvalues of the strings. The eigenvalue at site k is underlined ($\underline{k}$) if the site corresponds to an LZ77 cut, while it has a hat ($\hat{k}$) in case of an LZ78 cut.

$$\begin{gathered}\text{Eigensequence of } S_1 : \\ \underline{\hat{1}}\ \underline{\hat{2}}\ 2\ \hat{2}\ \underline{2}\ 2\ \hat{2}\ 2\ \hat{2}\ \underline{2}\ 2\ \hat{2}\ 2\ 2\ 2\ \hat{2}2\ 2\ 2\ 2 \\ \text{Eigensequence of } S_2 : \\ \underline{\hat{1}}\ \underline{\hat{2}}\ 2\ \underline{\hat{2}}\ 2\ \underline{\hat{5}}\ 6\ \underline{\hat{6}}\ 7\ 7\ \hat{8}\ \underline{10}\ 10\ 10\ \hat{1}0\ \underline{10}\ 13\ 14\ 15\ \underline{\hat{1}7}\end{gathered} \tag{3}$$

In either strings S_1 and S_2, both LZ77 and LZ78 show eigensequences that are not strictly increasing. Therefore, neither LZ77 nor LZ78 parsing is innovative.

(*iii*) By definition given in Theorem 1.3.1 (*i*), a history step $x(h_i, h_{i+1})$ is primitive if $h_{i+1} \doteq \min\{h > h_i \mid \lambda_x(h) > \lambda_x(h_i)\}$. Therefore, primitive parsing is the longest innovative parsing because it takes into account any possible innovative step and each primitive step is the shortest possible innovative step.

□

Remark 1.4.1. *Notice that from properties* (*ii*) *and* (*iv*) *of theorem1.2.1 follows that, among all possible histories of a given string* x*, the exhaustive parsing is the parsing into maximally long production steps, whereas from*

property (iii) of theorem1.4.1 follows that the primitive parsing is the parsing into minimally long production steps only among all possible innovative histories of x.

Indeed, we have

$$c_E(x) = \min_{\mathcal{H}\ history\ of\ x} c_{\mathcal{H}}(x) = \min_{\substack{\mathcal{H}\ \textbf{innovative} \\ history\ of\ x}} c_{\mathcal{H}}(x)\ .$$

In this sense the primitive parsing is a refinement of exhaustive parsing, since

$$c_{prim}(x) = \max_{\substack{\mathcal{H}\ innovative \\ history\ of\ x}} c_{\mathcal{H}}(x)\ .$$

Corollary 1.4.1. *Let x and y belong to $\mathcal{A}^*$.*

If $Dic(x) \bigcap Dic(y) = \emptyset$ then

$$c(xy) = \begin{cases} c(x) + c(y) \\ \qquad \textit{if the last step of the exhaustive history of } x \textit{ is exhaustive} \\ c(x) + c(y) - 1 \\ \qquad \textit{if the last step of the exhaustive history of } x \textit{ is not exhaustive} \end{cases}$$

and

$$c_{prim}(xy) = \begin{cases} c_{prim}(x) + c_{prim}(y) \\ \qquad \textit{if the last step of the primitive history of } x \textit{ is primitive} \\ c_{prim}(x) + c_{prim}(y) - 1 \\ \qquad \textit{if the last step of the primitive history of } x \textit{ is not primitive} \end{cases}$$

Proof. First, notice that the assumption on the vocabularies is realistic for sequences built on alphabets with more than 3 letters (e.g. DNA sequences are written in $\{A, C, G, T\}$ alphabet and wide parts of them are separately binary).

Now, for each $1 \leqslant h \leqslant l(y)$, let $xy(1,h)$ denote the joint sequence made of x followed by $y(1,h)$. The dictionary of this joint sequence may be written as

$$\begin{aligned} \mathrm{Dic}(xy(1,h)) &= \mathrm{Dic}(x) \cup \mathrm{Dic}(y(1,h)) \cup \\ &\quad \cup \{x(j,l(x))y(1,m) \mid 1 \leqslant j \leqslant l(x) \text{ and } 1 \leqslant m \leqslant h\} . \end{aligned}$$

Since by assumption $\mathrm{Dic}(x)$ and $\mathrm{Dic}(y(1,h))$ are disjoint, then for every $1 \leqslant j \leqslant l(x)$ and $1 \leqslant m \leqslant h$, the word $x(j,l(x))y(1,m)$ does not belong to $\mathrm{Dic}(x) \cup \mathrm{Dic}(y(1,h))$. For instance, if $x(j,l(x))y(1,m) \in \mathrm{Dic}(x)$ then $y(1,m) \in \mathrm{Dic}(x)$, which may not happen.

Hence, the eigenvocabulary of the joint sequence is

$$\begin{aligned} \lambda\mathrm{Dic}(xy(1,h)) &= \mathrm{Dic}(xy(1,h)) \setminus \mathrm{Dic}(xy(1,h-1)) = \\ &= \lambda\mathrm{Dic}(y(1,h)) \cup \{x(j,l(x))y(1,h) \mid 1 \leqslant j \leqslant l(x)\} . \end{aligned}$$

Consequently, the eigenvalue of the joint sequence $xy(1,h)$ equals

$$\#\lambda\mathrm{Dic}(xy(1,h)) = \lambda_y(h) + l(x) \tag{4}$$

for every $1 \leqslant h \leqslant l(y)$.

Now the theses may be easily derived thanks to Theorem 1.3.1, as follows.

We shall adopt the following notations, also to be consistent with the above formula (4):

- the sites of the exhaustive or primitive cuts in the joint sequence xy are $h_1, \ldots, h_{c(xy)}$;
- the sites of the exhaustive or primitive cuts in the individual sequence x are $k_1, \ldots, k_{c(x)}$;

– the sites of the exhaustive or primitive cuts in the individual sequence y are $\ell_1, \ldots, \ell_{c(y)}$.

The proof shall be shown in several steps.

(a) The last step of the exhaustive parsing of x is exhaustive.

For $i = 1, \ldots, c(x)$ the cut sites of the joint sequence coincide with the cut sites of x: $h_i = k_i$.

We now show by induction that for $i \geqslant c(x) + 1$ the cut sites of the joint sequence also coincide with the cut sites of y in the sense that: if $i = j + c(x)$ then $h_i = \ell_j + l(x)$. The shift is necessary because the sites for y start from $\ell_1 = 1$, while y_1 is the $(l(x) + 1)$-th symbol in the joint sequence. The first step is trivial.

By definition of Theorem 1.3.1, when $i \geqslant c(x)+1$ for the primitive cut h_i we have

$$\begin{aligned} h_i &= \min\{h \mid \lambda_{xy}(h) > h_{i-1}\} = \\ &\text{use formula (4)} \\ &= \min\{h \mid \lambda_y(h - l(x)) + l(x) > h_{i-1}\} = \\ &\text{use inductive hypothesis: } h_{i-1} = \ell_{j-1} + l(x) \\ &= \min\{h \mid \lambda_y(h - l(x)) + l(x) > \ell_{j-1} + l(x)\} = \\ &= \ell_j + l(x) \ . \end{aligned} \tag{5}$$

(b) The last step of the exhaustive parsing of x is not exhaustive.

The cut sites of xy coincide with the cut sites of x only up to the $(c(x)-1)$-th step. Hence, due to the fact that the dictionaries of x and y are disjoint, the $c(x)$-th cut site is always in y_1. The following steps coincide with the cut sites in pure y and this may be proved as above.

(c) The last step of the primitive parsing of x is primitive.

The proof is the same as in case (a), with the exception of formula (5), that changes as follows.

$$\begin{aligned} h_i &= \min\{h \mid \lambda_{xy}(h) > \lambda_{xy}(h_{i-1})\} = \\ &\text{use formula (4)} \\ &= \min\{h \mid \lambda_y(h - l(x)) + l(x) > \lambda_y(h_{i-1} - l(x)) + l(x)\} = \\ &\text{use inductive hypothesis: } h_{i-1} = \ell_{j-1} + l(x) \\ &= \min\{h \mid \lambda_y(h - l(x)) > \lambda_y(\ell_{j-1})\} = \\ &= \ell_j + l(x) \ . \end{aligned} \tag{6}$$

(d) The last step of the primitive parsing of x is not primitive.

The proof is the same as in case (b).

□

Remark 1.4.2. *The primitive parsing is not subadditive on* $\mathcal{A}^*$

Proof. We show here two strings for which is possible to see that the primitive parsing is not subadditive. Given $S_1 = 101011$ and $S_2 = 1001$ we have:

Eigenvalues S_1	1	2	2	2	2	5	
Sequence S_1	**1**|	**0**|	**1**	**0**	**1**	**1**|	$c_{prim}(S_1) = 3$
Eigenvalues S_2	1	2	2	3			
Sequence S_2	**1**|	**0**|	**0**	**1**|			$c_{prim}(S_2) = 3$

where with the symbol | we underline the positions where the cuts happen. When we append the two sequences, we obtain:

Eigenvalues $S_1 \wedge S_2$	1	2	2	2	2	5	5	6	8	8	
Sequence $S_1 \wedge S_2$	**1**$\|$	**0**$\|$	**1**	**0**	**1**	**1**$\|$	**1**	**0**$\|$	**0**	**1**$\|$	$c_{prim}(S_1 \wedge S_2) = 5$

Eigenvalues $S_2 \wedge S_1$	1	2	2	3	4	4	5	6	6	7	
Sequence $S_2 \wedge S_1$	**1**$\|$	**0**$\|$	**0**	**1**$\|$	**1**$\|$	**0**	**1**$\|$	**0**$\|$	**1**	**1**$\|$	$c_{prim}(S_2 \wedge S_1) = 7$

Then, we have that $7 = c_{prim}(S_2 \wedge S_1) > c_{prim}(S_1) + c_{prim}(S_2) = 6$

□

1.5 Optimality of the Exhaustive Parsing

In this section we want to report a central result of [72] about the convergency of the exhaustive entropy to the entropy.

Theorem 1.5.1. *For an ergodic source with positive entropy h and for every $\varepsilon > 0$ it holds:*

$$\lim_{n \to +\infty} Prob\left\{ c(x^n) < \frac{nh}{\log(n)}(1-\varepsilon) \right\} = 0 \ .$$

Proof. We follow the proof of the analogous theorem in [72], which was valid assigning the same probability measure $\frac{1}{|\mathcal{A}|^n}$ to each element of $\mathcal{A}_n$..

Let $[r]^-$ denote the largest integer not exceeding the real number r and $[r]^+$ denote the least integer not smaller than r.

Consider $x \in \mathcal{A}^{\mathbb{N}}$. For its truncated x^n, consider a parsing

$$L(x^n) = x(1,\ell)x(\ell+1,2\ell)\dots x(k\ell+1,n)$$

of x^n where all words (but at least the last one) have length $1 \leqslant \ell \leqslant n$ and $k = [\frac{n}{\ell}]^-$. Let

$$\delta_{x^n}(i) = \begin{cases} 0 & \text{if } x(1,(i-1)\ell) \to x(1,i\ell) \\ \\ 1 & \text{otherwise} \end{cases}$$

and let $x(1, \ell_i)$ be the sequence obtained in a single exhaustive production step from $x(1, (i-1)\ell)$.

Let now compare the word (or block) obtained from an exhaustive production step respect to the word obtained with an uniform segmentation;

If $\delta_{x^n}(i) = 1$, then the block obtained is not reproducible from the previous block; this means:

a) that the block is equals to the previous block plus some new character;
or

b) that the whole block is different from the previous block.

Let $x^u(1, il)$ denote the block obtained with uniform parsing under the condition $\delta_{x^n}(i) = 1$.

Respect to the case a) the component obtained with the exhaustive parsing $x^E(1, l_i)$ can be equals (in the case that there is only one new character) or less than $x^u(1, il)$ (if there are more than one new character)

In the case b) the exhaustive component is surely shorter than $x^u(1, il)$, according to the rule construction of the exhaustive parsing (there is always a cut after a new character).

Example 1.5.1. *Let's try to explain this with an example: Let* $x = AATATTACTGHN$ *be a string. Suppose to divide it in blocks of length* $l = 3$

A A T	*A T T*	*A C P*	*G H N*
Block1	*Block2*	*Block3*	*Block4*

Block2, Block3, and Block4 are examples of the three cases blocks previously seen. In particular, the exhaustive parsing of the characters in Block2 (we suppose now that the Block1 is fixed) is equal to components obtained with the uniform parsing (case a) with only one new character):

Fixed	*ExhComponent*
A A T	*A T T*
Block1	*Block2*

If after the Block1 we had the Block3, the exhaustive parsing on the Block3 (Block1 always fixed) obtains two components:

Fixed	*ExhComponents*
A A T	*A C* | *P*
Block1	*Block3*

This is equivalent to the case a) where the uniform component have more than one new character.

Otherwise, if Block1 was followed by the Block4, we are in the case b) (all the character in the Block3 did not appeared previously) and three components are made up by the exhaustive parsing:

Fixed	*ExhComponents*
A A T	*G*| *H* | *N* |
Block1	*Block4*

Therefore, in both the two cases (case a) and case b)) the length of the exhaustive block l_i is less than the length of the uniform block l_i or, otherwise,

$$\delta_{x^n}(i) = 1 \text{ if and only if } il \geqslant \ell_i.$$

Hence

$$c(x^n) = c_E(x^n) \geqslant \sum_{i=1}^{k} \delta_{x^n}(i) \ .$$

Now we consider the expectation of the complexity over the ensemble $\mathcal{A}_n$ and obtain:

$$\begin{aligned} \overline{c(x^n)} &= \sum_{x^n \in \mathcal{A}_n} \mu(x^n) c(x^n) \geqslant \\ &\geqslant \overline{\textstyle\sum_{i=1}^{k} \delta_{x^n}(i)} = \textstyle\sum_{i=1}^{k} \overline{\delta_{x^n}(i)} \ . \end{aligned}$$

According to the well-known Shannon-Mc Millan theorem, for every $\varepsilon > 0$, there is $\delta > 0$ such that sequences of sufficiently large length n from an ergodic stationary source with alphabet size α and entropy $0 < h \leqslant 1$ may be partitioned into two particular subsets.

A first subset $\mathcal{T}$ contains "typical" sequences that are almost uniformly distributed: for each $x^n \in \mathcal{T}$

$$\Big|\frac{\log_\alpha(\mu(x^n))}{n} + h\Big| < \varepsilon \ .$$

A second subset $\mathcal{R}$ is made of "rare" sequences and it is negligible:

$$\sum_{x^n \in \mathcal{R}} \mu(x^n) < \delta \ .$$

Notice that

$$\begin{aligned}
\overline{\delta_{x^n}(i)} \ &= Prob\ \{x(1,(i-1)\ell) \nrightarrow x(1,i\ell)\} = \\
&= 1 - Prob\ \{x(1,(i-1)\ell) \to x(1,i\ell)\} \geqslant \\
&\geqslant 1 - \sum_{j=1}^{(i-1)\ell} Prob\ \{x((i-1)\ell+1,i\ell) = x(j,\ell+j-1)\}
\end{aligned}$$

Now, if an ℓ-long word $x((i-1)\ell+1,i\ell)$ is rare, then $Prob\ \{x((i-1)\ell+1,i\ell) = x(j,\ell+j-1)\}$ is negligible. Otherwise, since typical words are almost uniformly distributed, then at least definitely for $\ell \to +\infty$ we have that

$$Prob\ \{x((i-1)\ell+1,i\ell) = x(j,\ell+j-1)\} = e^{-\ell h} + \varepsilon_1$$

where for the Shannon-Mc Millan theorem the probability of a typical sequence of length ℓ is almost $e^{-\ell h}$

From which

$$\overline{\delta_{x^n}(i)} \geqslant 1 - \ell(i-1)e^{-\ell h} + \varepsilon_2 \qquad \forall\ i \ . \tag{7}$$

Hence

$$\begin{aligned}
\overline{c(x^n)} &\geqslant \sum_{i=1}^{k} \overline{\delta_{x^n}(i)} \geqslant \\
&\geqslant \sum_{i=1}^{k} O(1-\ell(i-1)e^{-\ell h}) = \\
&= O\left(k - \ell e^{-\ell h}\frac{k(k-1)}{2}\right) = \\
&\text{by definition, } k = [\tfrac{n}{\ell}]^- \sim \tfrac{n}{\ell} \text{ for big } n \text{ and } \ell \\
&= \tfrac{n}{\ell} - \tfrac{n}{2}\left(\tfrac{n}{\ell}-1\right)e^{-\ell h} = \\
&= (\tfrac{n}{l}-1) - \tfrac{n}{2}\left(\tfrac{n}{\ell}-1\right)e^{-\ell h} + 1 \\
&= (\tfrac{n}{l}-1)(1-\tfrac{n}{2}e^{-\ell h}) + 1 \\
&\geqslant \left(\tfrac{n}{\ell}-1\right)\left(1-\tfrac{n}{2}e^{-\ell h}\right)
\end{aligned}$$

If we choose:

$$\ell = \left[\frac{1}{h}(\log(n) + \log\log(n))\right]^+$$

we may conclude that:

$$\overline{c(x^n)} \geqslant \frac{nh}{\log(n)}\left(1 - O\left(\frac{\log\log(n)}{\log(n)}\right)\right) \tag{8}$$

As already remarked by the authors of [72], by a slight modification of the proof of their Theorem 2 in the light of Shannon-Mc Millan Theorem, it may be easily proved that

$$\overline{c(x^n)} \leqslant \frac{hn}{\log(n)} \tag{9}$$

The last inequalities (8) and (9) imply the conclusion of this proof. □

This proof is essentially based on the count of the average number of the parsing blocks, and this count is done comparing with the average number of the uniform parsing blocks.

Remark 1.5.1. *A result analogous to Theorem 1.5.1 is also true when the complexity is that of LZ77 (see Theorem II.2.1 from [97]):*

$$c_{77}(x^n)\log(n) \longrightarrow nh \quad \textit{for } n \to +\infty$$

almost surely when x drawn from an ergodic process with entropy h.

An alternative proof of the optimality (related to the Dinamical System Theory) is obtained by Ornsteinn and Weiss. They show an interesting connection between entropy and partitions of a finite-alphabet stationary process into variable-length blocks; in [88] they prove two theorems concerned with two different kind of parsing:

a) parsings where the blocks are pairwise distinct

b) parsings where each block has already been "essentially" seen somewhere to the left

The first teorem (Theorem A) shows that eventually almost surely, for any partition into distinct words, most of the sample path is covered by blocks whose length is *at least* $\frac{\log n}{h}$. The second theorem (Theorem B) shows that for any partition into words that have been seen in the past, most of the sample path is covered by blocks whose length are *at most* $\frac{\log n}{h}$.

This second theorem is proved also in the case that each block occurs somewhere to the left only after a fixed number of symbols at the beginning and the end of the block have been deleted.

So, respect to a parsing $c(x^n) \sim \frac{n}{l_{max}}$, from theorem A it is possible to prove that

$$\limsup_{n\to\infty} \frac{c(x^n)\log n}{n} \leqslant h \text{ a.s}$$

while, from theorem B:

$$\liminf_{n\to\infty} \frac{c(x^n)\log n}{n} \geqslant h \text{ a.s}$$

Now, from these two theorems, *any* parsing that satisfy the two condition a) and b), will have most of its blocks with lengths approximately equal to $\frac{\log n}{h}$.

In the particular case of theorem B where is not required that the blocks lie completely to the left, the lempel ziv parsing (exhaustive parsing but also the LZ77 parsing and other varianti) satisfies both condition a) and b) and

it is possible to apply the combination of the two theorem to the Lempel-Ziv parsing, as decribed by the theorem 4 in [88]:

Theorem 1.5.2. *If $\{x^n\}_1^\infty$ is a finite valued stationary ergodic process with entropy h, almost surely for an output $\{\eta^n\}_1^\infty$, given $\varepsilon > 0$, for all $n \geqslant N_\varepsilon$ and any "old" parsing of $\eta_1 \dots \eta_n$ into distinct blocks I_i, we have that*

$$\sum_{I_i : \frac{1}{h+\varepsilon} \log n \leqslant |I_i| \leqslant \frac{1+\varepsilon}{h} \log n} |I_i| \geqslant (1-\varepsilon)n \tag{10}$$

This theorem can get upper and lower bounds on the number of intervals in the parsing, but the information that it gives is more precise.

Chapter 2

Similarity metrics

In this chapter we analyze similarity metrics related to different teoretical approaches: the first part of the chapter concerns the so called "Information Distances" based on the Kolomogorov complexity. The second part analyzes the distances related to the Relative Entropy (or Kullback Leibler divergence), an indicator of the discrepancy between two different sources.
Due to the uncomputability of both two theoretical distances, different approximation methods are developed; here we show those related to the compressor theory, parsing and the BWT transformation.

2.1 Information Distance

The Kolmogorov complexity, or algorithmic entropy, $K(x)$ of a string x is the length of the shortest binary program x^* to compute x on a universal computer (such as a universal Turing machine) [77]. Thus, $K(x) = |x^*|$, the length of x^*, denotes the number of bits of information from which x can be computationally retrieved. This way of seing the complexity of a string as absolute quantification of the amount of information in it, leads to a theory of absolute information contents of individual objects in contrast to classical information theory of Shannon Entropy wich deals with average information to transmit objects produced by a random source. With this definition of complexity of strings, it is possible to study the question of an information distance metric between individual objects, rather than the

measure of the statistical remoteness between two stocastic sources through the approximation of the relative entropy

We need a notion of distance that takes into account the problem of clustering objects. Numerous and non equivalent distances have been developed in the statistical field (see for example [115]), or following other approaches include various types of edit distances between pairs of strings (Hamming distance [46]). *A priori* it is not obvious what is the "best" measure of distance between two strings. "Best" could be intended as the minimal amount of energy needed to traslate between x and y.

The answer to this request is the *Information Distance*, formally defined as the length $E(x, y)$ of a shortest binary program that computes x from y (as well as computing y from x.)
What quantities defines this Information Distance? In [4] it is shown that $E(x, y) = \max\{K(y|x), K(x|y\}$ up to an additive $O(\log \max\{K(y|x), K(x|y)\})$ term, where $K(x|y)$ is the conditional Kolomogorov complexity, later defined. We resume now the main steps of their arguments.

Definition 2.1.1 (Distance and Metric)**.** *A distance is a function D with non negative values, defined on the Cartesian product $\mathcal{X} \times \mathcal{X}$ of a set $\mathcal{X}$. It is called a metric on x if for every $x, y, z \in \mathcal{X}$:*

- $D(x, y) = 0$ *iff* $x = y$ *(the identity axiom);*
- $D(x, y) = D(y, x)$ *(the symmetry axiom);*
- $D(x, y) + D(y, z) \geqslant D(x, z)$ *(the triangle inequality).*

Definition 2.1.2 (upper semi-computable function)**.** *A real valued function $f(x, y)$ is upper semi-computable if there exists a rational valued recursive function $g(x, y, t)$ such that*

i) $g(x, y, t + 1) \leqslant g(x, y, t)$*;*

ii) $\lim_{t\to\infty} g(x, y, t) = f(x, y)$

It is lower semi-computable if $-f(x,y)$ is upper semi-computable.

At last, $f(x,y)$ is computable if it is both upper and lower semi-computable.

Definition 2.1.3 (admissible distance)**.** *Let $\Omega = \{0,1\}^*$, $D : \Omega \times \Omega \to \mathbb{R}^+$ is an admissible distance if it is symmetric, upper semi-computable distance function and such that, for every pair of objects $x, y \in \Omega$, $D(x,y)$ is the length of a binary prefix code-word that is a program computing x from y, and vice-versa.*

From the definition D must satisfy the following *Kraft inequality* which gives some natural restriction on the density of the neighbor of any given $x \in \{0,1\}^*$:

$$\sum_{y \in \{0,1\}^*} 2^{-D(x,y)} \leqslant 1. \tag{1}$$

Remark 2.1.1. *It is easy to see that the function $K(x)$ and $K(y|x^*)$ are upper semi-computable, but they are not computable.*

Morever, it is possible to see that $K(y|x)$ has the property that for each x

$$\sum_{y} 2^{-K(y|x)} \leqslant 1.$$

Indeed, when we use the $K(y|x)$ we are building a subset of the length set of a prefix-code (or instantaneous code), and we know that the codeword lengths of prefix-code must satisfy the Kraft inequality (1) *(vice-versa: for the set of codewords lengths that satisfy the Kraft inequality it does exist a prefix-code with these word lenghts) [25].*

Finally we observe that, with respect to this normalization property, the conditional complexity $K(x|y)$ is minimal. That is, for every upper semi-computable function $f(x,y)$

$$K(y|x) \overset{+}{<} f(x,y).$$

(with $\overset{+}{<}$ we denote an inequality to within an additive constant)

Following the idea of identification of the Kolmogorov complexity $K(x)$ as the information content of x we want to define the complexity of a string x

referred to a string y as the information distance. This brings to, the need of introducing the *conditional descriptional complexity* $K_F(y|x) := \min\{l(p) : F(p,x) = y\}$, where with F we denote a partial recursive function computed by a prefix Turing machine. Denoted with U the universal self-delimiting Turing machine with the property that $K_U(y|x) \leqslant K_F(y|x) + c_F$, with c_F a constant depending on F, we write

$$K(y|x) := K_U(y|x)$$

and call $K(y|x)$ the *conditional Kolmogorov complexity* of x with respect to x.

After the preceding considerations, it seems natural to choose $K(y|x)$ as a possible canditate to satisfy our initial request of discovery the "best" distance. Unfortunately, the conditional complexity $K(y|x)$ is unsuitable as information distance because it is asymetric; this asymetry can be solved with the sum $K(y|x)+K(x|y)$. However, even in this case we have a problem: the resulting metric will overestimate the information required to traslate between x and y in case there is some redundancy between the information required to get to y from x and the information required to get to x from y.

Definition 2.1.4. *[4] The max distance between x and y is defined by*

$$E(x,y) := \max\{K(x|y), K(y|x)\}$$

In [4] it has been accurately proved that the $E(x,y)$ so defined is equal to the information distance $E_0(x,y) := \min\{l(p) : U(p,x) = y, U(p,y) = x\}$.

Clearly, $E(x,y)$ is simmetric and then, recalling that $K(\cdot|\cdot)$ satisfies both the upper semi-computable property and Kraft's inequality, we can say that $E(x,y)$ *is an admissible distance.*

Theorem 2.1.1. *[76] The Information Distance $E(x,y)$ is an admissible distance that satisfies the metric inequalities up to an additive constant, and it is minimal in the sense that for every admissible distance $D(x,y)$ we have $E(x,y) \leqslant D(x,y) + O(1)$. The $E(x,y)$ is defined as a* **universal admissible metric**.

Proof. How we have previously underlined, the nonnegativity and symmetry properties follow easily from the property of $K(y|x)$, as well as the normalization and the minimal property.

It remains to prove the triangle inequality: we assume, without loss of generality, that $E(x,z) = K(z|x)$. Then:

$$\begin{aligned} E(x,z) &= K(z|x) \stackrel{+}{<} K(y,z|x) \stackrel{+}{<} K(y|x) + K(z|x,y) \\ &\stackrel{+}{<} K(y|x) + K(z|y) \leqslant E(x,y) + E(y,z) \end{aligned}$$

□

An important observation at this point is that every specific feature of objects induces a distance, and every specific distance measure can be viewed as a quantification of an associated feature difference. The above theorem states that among all features that correspond to upper semicomputable distances, satisfing the density condition (1), the information distance is universal in that among all such distances it is always smallest up to constant precision. That is, *it accounts for the dominant feature in which two objects are alike.*

A useful consideration, at this point, could be that the notion of information content of individual objects can only be useful if the quantity of information is only an attribute of the object and is indipendent of the means of description [77].

2.1.1 Normalized Information Distance (NID)

Now, we have a definition of Information Distance $E(x,y)$ and we know, from Theorem 4.2 in [4], that this distance is also a metric, up to an additive fixed finite constant. This is an absolute distance, but, if we are interested in investigating the similarity of strings, then we are more interested in *relative distances*. Using the information distance now defined, indeed, it could happen that a sequence x results nearer to sequences y only because the length of y is almost equal to one of x.

We need, thus, of a notion of normalized distance:

Definition 2.1.5 (normalized distance or similarity distance)**.** *A normalized distance or similarity distance, is a function* $d : \Omega \times \Omega \to [0,1]$ *that is symmetric* $d(x,y) = d(y,x)$*, and for every* $x \in \{0,1\}^*$ *and every constant* $e \in [0,1]$

$$|\{y : d(x,y) \leqslant e \leqslant 1\}| < 2^{eK(x)+1} \tag{2}$$

Remark 2.1.2. *The density requirement* (2) *is implied by a "normalized" version of the Kraft inequality (Lemma IV in [76]):*

$$\sum_y 2^{-K(x)\, d(x,y)} \leqslant 1.$$

Finally, we can give the definition of Normalized Information Distance:

Definition 2.1.6. *Given two sequences* x *and* y*, define the function*

$$d_K(x,y) = \frac{\max\{K(x|y^*), K(y|x^*)\}}{\max\{K(x), K(y)\}} \tag{3}$$

(where x^* *and* y^* *are defined as in section 2.1)*

Remark 2.1.3. *As discussed in [76], there are several different alternatives for the denominator, but they turn out to be all wrong.*

In particular, if we divide by $\max\{|x|, |y|\}$ *the corresponding metric will not satisfy the triangle inequality and its universal property is lost.*

If instead it is used $K(x,y)$ *as denominator, then* $d_K(x,y) = 1/2$ *for any pair of independent and fully random sequences.*

Note that , if $K(y) \geqslant K(x)$ then we can write

$$d_K(x,y) \;=\; \frac{K(y) - I(x:y)}{K(y)} \;=\; 1 - \frac{I(x:y)}{K(y)}.$$

In [76] is proved that this d_K takes values in the range $[0, 1+O(\frac{1}{\max\{K(x),K(y)\}})]$ and it can be showed that it is a normalized distance (or similarity distance), that is, it can be proved the density condition of (2) [76]. Morever, this function satisfies the metric properties up to an additive precision $O(\frac{1}{K})$, where

K is the maximum of the Kolmogorov complexities of the objects involved in the equality. Then, this $d_K(x, y)$ is a **similarity metric**.

Finally we are interested in the fact that this NID is *universal* in the sense that every metric expressing similarity according to some feature, that can be computed from the objects concerned, is included (in the sense of minorized) by the universal metric. This should be understood in the sense that if two files are similar (that is, close) according to particular metric, then they are also similar (that is close) in the sense of the normalized information metric.

Theorem 2.1.2. *[76] The normalized information distance $d(x, y)$ minorizes every upper semi-computable normalized distance $f(x, y)$ by $d(x, y) \leqslant f(x, y)+ O(\frac{1}{K})$ where $K = \min\{K(x), K(y)\}$.*

For this universality property we call the function $d_K(x, y)$ as **"the"** similarity metric.

2.1.2 Normalized Compression Distance (NCD)

Now we want develop an analogue of the NID based on real world compressors, called the "Normalized Compression Distance" (NCD). The main issue related to this approximation is intrinsic factor of the notion of Kolmogorov complexity; the Kolmogorov complexity is indeed a noncomputable function, thus, $K(x)$ is, in last analysis, the lower bound of what a real world compressor can possibly achieve. It is very important to stress again that in any practical approximation of the Kolmogorov complexity $K(x)$ of a given string, we will not be able to determine how fast in the length of x our estimate converges to its true value, besides few asymptotic results for ergodic sources. Moreover, we will never consider or discuss the (usually divergent) time needed for our algorithm to produce a final output.

Definition 2.1.7. *The approximation of NID is called the Normalized Compression Distance (NCD) and is defined by:*

$$NCD(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}} \tag{4}$$

where with $C(xy)$ we denote the compressed size of the concatenation of x and y, $C(x)$ is the compressed size of x, and $C(y)$ that of y. The NCD is a non negative number $0 \leqslant NCD \leqslant 1 + \varepsilon$ and shows how much the two files are different.

Note that, while in [4, 76] only the boundary case $K(x) = C(x)$ has been discussed, a more general theory of normalized compression distance has been developed in [24] based on the notion of a *normal compressor*.

Definition 2.1.8 ([24])**.** *A compressor C is normal if the following axioms are satisfied up to an additive $O(\log n)$, where n is the maximal binary length of the elements involved in the inequalities:*

1. *Idempotency: $C(xx) = C(x)$, and $C(\varepsilon) = 0$, where ε is the empty string.*
2. *Monotonicity: $C(xy) \geqslant C(x)$.*
3. *Symmetry: $C(xy) = C(yx)$.*
4. *Distributivity: $C(xy) + C(z) \leqslant C(xz) + C(yz)$.*

Remark 2.1.4. *While the first two axioms, idempotency and monotonicity, are usually satisfied by any real-world compressor, the simmetry property is much more subtle: usually this property is not satisfied by Lempel-Ziv type compressors such as gzip and pkzip, and also the predictive PPM family, like PPMZ, are possibly not exactly symmetric. This fact is clearly related to the stream-like mechanism of these compressors. On the other hand, the block-coding like compressor, such as bzip2, implement a global analysis of the string (through the Burrows-Wheller transform) and are to a great extent symmetric.*

Finally, the distributivity properties listed here (the proof follow from the monotocity property) is a weaker form of a strong property satisfied by the Kolmogorov complexity $K(xyz) + K(z) \leqslant K(xz) + K(yz)$ and it is usually satisfied by real-word compressor, at least up to the required precision.

If $C(y) > C(x)$ the NCD become equal to:

$$NCD(x,y) = \frac{C(xy) - C(x)}{C(y)} = 1 - \frac{C(y) - C(y|x)}{C(y)} = 1 - \frac{I(x,y)}{C(y)}$$

Theorem 2.1.3. *[24] If the compressor is normal, then NCD is a similarity metric.*

Let's see now the NCD approximation in detail. The numerator of (3) can be rewritten as $\max\{K(x,y) - K(x), K(y,x) - K(y)\}$, within logarithmic additive precision by the additive property of Kolmogorov complexity (see [77]). The term $K(x,y)$ represents the length of the shortest program for the pair (x,y). The joint Kolmogorov complexity $K(x,y)$ of two sequences $x, y \in \{0,1\}^n$ is log-equivalent to their concatenated Kolmogorov complexity $K(xy)$. In order to prove this fact we need to introduce some definitions.

Definition 2.1.9. *A pairing function is any function $< \cdot, \cdot >: \mathbb{N} \times \mathbb{N} \to \mathbb{N}$. The pairing is said total if it is defined for each (x,y) and it is said prefix code if its range is a prefix-free subset of $\mathbb{N}$.*

We recall that every binary sequence may be read as a natural number via the standard mapping σ of $\{0,1\}^*$ onto $\mathbb{N}$ (see [77]). Therefore we can say that any pairing associates a binary string to a pair of binary strings:

$$(x,y) \in \{0,1\}^* \times \{0,1\}^* \mapsto \sigma^{-1} < (x,y) > \doteq \sigma^{-1} < \sigma(x), \sigma(y) > \in \{0,1\}^* \ .$$

If the pairing is total, one-to-one and it is a prefix code, then for any pair (x,y) of binary strings in $\{0,1\}^* \times \{0,1\}^*$ we can unambigously read the pair as a unique binary sequence, i.e. every prefix code is a uniquely decodable code. The converse is not true (see [42]).

Theorem 2.1.4. *Total and one-to-one pairings that are also prefix codes do exist.*

Proof. Let $E : \mathbb{N} \longrightarrow \{0,1\}^*$ be a total, one-to-one, prefix code. For instance, let $M : \{0,1\}^* \longrightarrow \{0,1\}^*$ s.t. $M(x) \doteq 1^{l(x)}0x$. Let $\hat{M}(x) \doteq M(l(x))x$, where $l(x)$ is interpreted as a binary string according to the standard identification

of $\mathbb{N}$ to $\{0,1\}^*$. Then E defined as $E(\hat{M}(x)) = x$ is a total, one-to-one prefix code, when the binary string is interpreted as a natural number.

Then the mapping defined by $< x, y > \doteq E(x)E(y)$ is a total (trivial) and one-to-one pairing and it is a prefix code.

□

Now, we are ready to prove that joint Kolmogorov complexity $K(x,y)$ of two sequences $x,\ y \in \{0,1\}^n$ is log-equivalent to their concatenated Kolmogorov complexity $K(xy)$, in the following sense.

Theorem 2.1.5. *$K(x,y) = K(xy)$ up to a logarithmic term.*

Proof. By definition, $K(x,y) \doteq K(< x, y >)$ where $< \cdot, \cdot >$ is a standard invertible prefix code pairing. Intuitively, the prefix code E defined in Theorem 2.1.4 may be used to define both a concatenation of x and y and a product string (x,y), as follows.

$$\text{Concatenation program:} \begin{cases} \textit{Write } x\textit{, for instance via prefix code } E \\ \\ \textit{Write } y\textit{, for instance via prefix code } E \end{cases}$$

$$\text{Product program:} \begin{cases} \textit{Write } x\textit{, for instance via prefix code } E \\ \\ \textit{newline or similar separator after } l(E(x)) \textit{ written bits} \\ \\ \textit{Write } y\textit{, for instance via prefix code } E \end{cases}$$

Since $l(E(x)) = l(x) + 2\log(l(x)) + 1$, then any two minimal programs defining either the concatenation or the product of sequences differ from about $\log(l(E(x))) = O(\log(l(x)))$. □

Corollary 2.1.1. *For any $x,\ y \in \{0,1\}^n$*

$$K(x,y) = K(y,x)$$

up to a logarithmic term of the order of $\max\{\log(l(x)), \log(l(y))\}$. □

Summarizing: from the theory: $K(x,y) = K(xy) = K(yx)$. But, when we deal with real-life compressors, the quantity $C(xy)$ is different from $C(yx)$; we take the smaller of the two because this is clearly the closest approximation to $K(x,y)$. Then:

$$\begin{aligned} \max\{K(x,y) - K(x), K(x,y) - K(y)\} &= K(xy) - \min\{K(x), K(y)\} \\ &\approx \min\{C(xy), C(yx)\} - \min\{C(x), C(y)\} \\ &\approx C(xy) - \min\{C(x), C(y)\} \end{aligned} \tag{5}$$

where with $C(xy)$ we denote the $\min\{C(xy), C(yx)\}$ (from the previous observation the compressors are not always simmetric, unless C is a normal compressor.)

As we have seen, if C is a normal compressor then $d_C(x,y) + O(1)$ is an *admissible distance*, where

$$d_C(x,y) \;=\; C(xy) - \min\{C(x), C(y)\}.$$

If, instead of the result of some real compressor, we substitute the Kolmogorov complexity for the lengths of the compressed files in the NCD formula, the result is a similarity metric, and it is called *Kolmogorov metric*. Note that when we use the real compressor we deal with sequence of finite length and the claimed universality of the Kolmogorov metrics holds only for indefinitely long sequences x, y. Moreover, there is always the problem that the Kolmogorov complexity is not computable, so we cannot know how well we are doing using the NCD, because we cannot compute how far the NCD is from the Kolmogorov metric.

While better compression of a string will always approximate the Kolmogorov complexity better, this may not be true for the NCD, because the formula involves subtraction and division, and the improvement for single items is not necessarily the same for all items.

Moreover, the approximation of the quantity $K(x,y) = K(xy)$ with the compression of two appended sequences $C(xy)$ is not proved, it is an heuristically approximation, assumed as a necessary condition for having the uni-

versality of NCD;[1] so the NCD is *quasi universal metric* in the sense that $NCD(x, y)$ minorizes any metric $d(x, y)$ under a strong condition given by the following theorem:

Theorem 2.1.6. *Let d be a computable similarity metric. Given a constant $a \geqslant 0$, let objects x, y be such that $C(xy) - K(xy) \leqslant a$. Then $NCD(x, y) \leqslant d(x, y) + \frac{(a+O(1))}{k}$ where $k = \max\{C(x), C(y)\}$*

Beside this approximation of NID, we recall also another compression distance that is used on DNA sequences to compare genomes and construct evolutionary trees. In [16], [75] is defined a distance:

$$d_{GenCompress}(x, y) = 1 - \frac{K(x) - K(x|y)}{K(xy)} \tag{6}$$

that is the same normalized information distance defined in equations $(V.1)$ and $(V.2)$ of [76]. Using the definition of mutual information, it is possible to rewrite (6) as:

$$d_{GenCompress} = 1 - \frac{I(y : x)}{K(xy)}.$$

This distance satisfies the triangle inequality, up to a small error term, and universality, but only within a factor of 2 (see for proofs [75]). In [16] $K(x)$ is heuristically approximated with $Compress(x)$, where $Compress$ indicates the use of compression algorithm GenCompress, and use $Compress(x|y)$ to heuristically approximate $K(x|y)$. For counting the conditional kolmogorov complexity, it is needed to convert the GenCompress program into the conditional version.

Now we introduce an approximation of the NID whit another tool that is different from compressor approach.

2.1.3 LZ-metric

The metric that will be shown now, was discussed for the first time in [90]. It can be seen as another approximation of the Information Distance above

[1] this problem about the compression of two appended sequences is faced in section 2.2.4

discussed but instead of the compressor as in [24], the approximator is the exhaustive parsing defined in section 1.2.

The idea that moves us to considering the parsing for a construction of a distance between sequences is the following: given two sequences x and y consider the sequence xy and its exhaustive history. Rewriting the subadditivity property of the ehxaustive parsing ($c(xy) \leqslant c(x) + c(y)$) as $c(xy) - c(x) \leqslant c(y)$, we say that the number of component needed to build y when concatenated to x will be lower than or equal to $c(y)$ and this, in turn, would reduce the number of exhaustive components.

Clearly, the more the two sequences x and y are similar, the lower the quantity $c(xy) - c(x)$ is referred to $c(y)$. How great is the difference will depend on the degree of similarity between x and y. Given a third string z, if a sequence x is closer to z than to y then we would expect $c(xz) - c(x)$ to be smaller than $c(xy) - c(x)$.

Example 2.1.1. *Let* $x = ATCATGCTAGTACGT$, $y = AATCGTATACGTCG$ *and* $z = AGTACGTTCATGCT$ *The exhaustive histories of these sequences are:*

$$H_E(x) = T.C.A.TG.CT.AG.TAC.GT$$
$$H_E(y) = A.AT.C.G.TA.TAC.GTC.G$$
$$H_E(z) = A.G.T.AC.GTT.CA.TG.CT$$

yelding $C(x) = C(y) = C(z) = 8$. *The exhaustive histories of the appended sequences* XY *and* YZ *are:*

$$H_E(xy) = T.C.A.TG.CT.AG.TAC.GT.|AA.TC.GTAT.ACGTC.G$$
$$H_E(xz) = T.C.A.TG.CT.AG.TAC.GT.|AGTACGTT.CATGCT$$

where $C(xy) = 13$ *and* $C(xz) = 10$. *Note that it takes two steps to build* z *from* xz *while five steps are necessary to generate* y *from* xy. *The reason is because* z *and* x *have more common patterns than* x *and* y.

The quantity $c(xy) - c(x)$ is therefore an indicator of the similarity between the building rules of the two strings. On this indicator it is possible to construct a distance measure between sequences.

Four distances are proposed from [90], each using the notion of complexity of LZ. We recall only the normalizated versions of these:

Definition 2.1.10. *For any $x, y \in \mathcal{A}^*$ we define the LZ-metric*

$$d_{LZ}(x,y) = \frac{\max\{c(xy) - c(x), c(yx) - c(y)\}}{\max\{c(x), c(y)\}} \tag{7}$$

For approximating the information distance defined in equations $(V.1)$ and $(V.2)$ of [76] the authors use:

$$d^*_{\mathrm{LZ}}(x,y) = \frac{c(xy) - c(x) + c(yx) - c(y)}{\frac{1}{2}[c(xy) + c(yx)]} \tag{8}$$

The function $d_{\mathrm{LZ}}(\cdot,\cdot)$ (and also $d^*_{\mathrm{LZ}}(x,y)$)is a distance metric. Indeed, by the definition, $d_{\mathrm{LZ}}(\cdot,\cdot)$ satisfies the simmetry property. The identity condition is satisfied up to an error term of $O(\frac{1}{c(x)})$. For the proof of the triangle inequality is needed the following Lemma:

Lemma 2.1.1.

$$c(xy) - c(x) \leqslant c(xz) - c(x) + c(zy) - c(z) \tag{9}$$

that, in turn, uses the following property:

$$c(xyz) - c(xy) \leqslant c(yz) - c(y) \tag{10}$$

This last equation says that the number of components z would have, if parsed using xy, in less than the number of components z would have, if parsed using y. Infact, we know, from the subadditivity that:

$$\begin{aligned} c(xyz) - c(xy) &\leqslant c(z) \\ c(yz) - c(y) &\leqslant c(z) \end{aligned}$$

then

$$\begin{aligned} c(xyz) - c(xy) - c(yz) + c(y) &\leqslant 0 \\ c(xyz) - c(xy) &\leqslant c(yz) - c(y) \end{aligned}$$

Noting (10) and that

$$c(xz) - c(x) \leqslant c(xyz) - c(x) \tag{11}$$

we obtain

$$c(xz) - c(x) \leqslant c(xyz) - c(x) \leqslant c(yz) - c(z) + c(xy) - c(x)$$

that is the proof of lemma 2.1.1.

Note that if we replace c with K we obtain

$$\begin{aligned} K(xyz) - K(xy) &= K(z|x,y) \\ &\approx K(x,z|y) - K(x|y) \\ &\leqslant K(x,z|y) \leqslant ?K(z|y) \end{aligned}$$

The proof of the triangle property of the formula (7) follows the proof of triangle property for the NCD metric and it is given in [90] using the subadditivity of the exhaustive parsing. We write the proof in the case of the formula (8), that is the formula used in our experiments as well. We can note that by definition the (8) satisfies the symmetry condition (up to a logarthmic factor) while the identity factor is satisfied up to an additive error term of $O(\frac{2}{c(S)})$ dependig on wheter the last component of the sequence S is exhaustive or not (if the last component is exhaustive the parsing of x appended to itself is equal to the parsing of x plus a component equal to the whole string x previoulsy seen). To prove the triangle inequality for (8) it is sufficient to show the two inequalities (the quantity $\frac{1}{2}$ in the denominator of (8) is omitted in this proof):

$$\begin{aligned} \frac{c(xz) - c(x)}{c(xz)} &\leqslant \frac{c(xy) - c(x)}{c(xy)} + \frac{c(yz) - c(y)}{c(yz)} \\ \frac{c(zx) - c(z)}{c(xz)} &\leqslant \frac{c(yx) - c(y)}{c(xy)} + \frac{c(zy) - c(z)}{c(yz)} \end{aligned}$$

we prove the first one only (the second is simmetric to the first one). Let $\delta = c(yz) - c(y) + c(xy) - c(x) - (c(xz) - c(x))$; this quantity is positive for the lemma (2.1.1), then we can write:

$$\begin{aligned} \frac{c(xz) - c(x)}{c(xz)} &\leqslant \frac{c(xz) - c(x) + \delta}{c(xz) + \delta} \\ &= \frac{c(xy) - c(x) + c(yz) - c(y)}{c(xy) + c(yz) - c(y)} \\ &\leqslant \frac{c(xy) - c(x)}{c(xy)} + \frac{c(yz) - c(y)}{c(yz)} \end{aligned}$$

since $c(xy) + c(yz) - c(y) \geqslant {}^1 c(xy) + c(y) - c(y) \geqslant c(xy)$ and $c(xy) + c(yz) - c(y) \geqslant c(xy) + c(y) - c(y) \geqslant c(yz)$ As the second inequality is proved symmetrically we have proved the triangle inequality.

2.2 The Relative Entropy

In this section we want to analyze distances between distributions. We recall the notion of the Relative Entropy (or Kullback-Leibler divergence) for two markovian sources. In realistic situations, namely in working with concrete finite realizations of ergodic unknown sources with unknown memory length, we face the pratical problem of estimating the Kullback-Leibler divergence.

In this chapter we show different approaches to estimate the divergence between two sources given the sequences: one using the parsing procedure, and another using the BWT transformation. Finally, we show a different type of distance, named n-gram distance, based on the n-gram frequencies of the two sequences.

[1] this follows for the property that says that $c(y) \leqslant c(yz)$. guarda dove l'hai messa e come si chiama

2.2.1 The Relative Entropy between two Markov sources

Given two positive probability mass function $p(x)$ and $q(x)$, the Relative Entropy (or the Kullback-Leibler Divergence) is defined as follows:

$$\mathcal{D}(q \parallel p) = \frac{1}{n} \sum_{x \in \mathcal{X}} q(x) \log \frac{q(x)}{p(x)} \tag{12}$$

When the sequence $\mathbf{x}$ is a sequence emitted from an unknown lth-order stationary Markov process $p(\cdot)$ over a finite alphabet $\mathcal{A}$, we know that the probability mass function p is defined as:

$$p(\mathbf{x}) = p(x_1^l) \prod_{i=l+1}^{n} p(x_i | s_{i-1}) \tag{13}$$

where $s_i = x_{i-l+1}^i = (x_{i-l+1}, x_{i-l+2}, \ldots, x_i)$ for $i \geqslant l$ and $s_i = (s_0, x_1, x_2, \ldots, x_i)$ for $i < l$, s_0 being the initial state.

Now, let $p(\cdot)$ and $q(\cdot)$ be two Markov probability measure, each of order no larger than same positive integer l.

Let $\mathbf{x}$ be a realization of $p(.)$ and let $\mathbf{z}$ be a realization of $q(.)$

We have:

$$\begin{aligned} \sum_{\mathbf{x} \in \mathcal{A}^n} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})} &= \sum_{a_1^l \in A^l} q(a_1^l) \log \frac{q(a_1^l)}{p(a_1^l)} \\ &\quad + (n-l) \sum_{a \in A; s \in A^l} q(s,a) \log \frac{q(a|s)}{p(a|s)} \end{aligned}$$

where the state s is the previous l letters.

Then, the Relative Entropy between these two mass functions is defined in the following way:

$$\begin{aligned} \mathcal{D}(q \parallel p) &= \mathcal{D}^l(q \parallel p) = \limsup_{n \to \infty} \sum_{\mathbf{x} \in \mathcal{A}^n} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})} \\ &= \sum_{a \in A; s \in A^l} q(s,a) \log \frac{q(a|s)}{p(a|s)} \end{aligned} \tag{14}$$

while the entropy respect to q is:

$$\begin{aligned} H(q) &= -\sum_{\mathcal{A};\mathcal{A}^l} q(s,a) \log \frac{q(s,a)}{q(s)} \\ &= -\sum_{\mathcal{A};\mathcal{A}^l} q(s,a) \log q(a|s) \end{aligned} \tag{15}$$

Rougly speaking, the relative entropy measures the distance between two different distributions, estimating the inefficiency of assuming that the distribution of a sequence $\mathbf{x}$ is p when the true distribuition is q. It is not a true distance in mathematical sense (it is not simmetric and does'nt satisfy the triangle inequality), but it is a usefull indicator for estimating the statistical differences between sequences belonging to different sources [25].

The estimate of the distribution of the source from a single realization $\mathbf{x}$ is a computational hard problem, so we need to approximate the quantities of the relative entropy.

2.2.2 The Merhav and Ziv theorem

In the chapter 1 we have seen an important result that connects the number of compenents of an exhaustive parsing to the entropy (see theorem 1.5.1).

Analogously, in [115] the authors define a kind of procedure that uses the parsing on a string to estimate the *cross entropy* $C(q \parallel p)$ between the realizations $\mathbf{z}$ and $\mathbf{x}$ of two sources with probability q and p respectively.

The cross entropy between two sources is defined as

$$C(q \parallel p) = -(D(q \parallel p) + H(q)) \tag{16}$$

(in the case of bernoullian sequences $C(q \parallel p) = q \log p + (1-q)\log(1-p)$)

The idea is to estimate the cross entropy term by building the parsing of $\mathbf{z}$ using the string $\mathbf{x}$. With $c(\mathbf{z}|\mathbf{x})$ the authors indicate the number of phrases obtained with the parsing procedure in $\mathbf{z}$ with respect to $\mathbf{x}$. Practically one has to find the largest integer m such that the sub-sequence $(z_1, \ldots, z_m) = (x_i, x_{i+1}, \ldots, x_{i+m-1})$ for some i. The string $(z_1, \ldots, z_m)$ is defined as the first

phrase of $\mathbf{z}$ with respect to $\mathbf{x}$. Then one starts from z_{m+1} and find, in a similar manner the longest sub-sequence $(z_{m+1}, \ldots, z_n)$ which appears in $\mathbf{x}$, and so on. The procedure ends once the entire sequence $\mathbf{z}$ has been parsed with respect to $\mathbf{x}$.

From the definition of Cross Entropy, it is evident that approximating this term we can approximate also the relative entropy (in section 1.5 we have seen, indeed, that the number of words obtained with the parsing of a string is an approximator of its entropy).

We resume brefly the steps in [115] that lead to the definition of the quantity that approximates the relative entropy.

From (12) (13) and (14) it is possible to obtain:

$$-\log p(\mathbf{z}) = nH(q) + nD(q \parallel p) \tag{17}$$

(the term $\log p(\mathbf{z})$ is equivalent to the cross entropy defined above).

Defined $Q(\mathbf{z} \parallel \mathbf{x}) = \frac{1}{n} c(\mathbf{z}|\mathbf{x}) \log n$, the main point of the article of Merhav and Ziv [115] is the proof of the following limit:

$$\lim_{n\to\infty} \left[-\frac{1}{n} \log p(\mathbf{z}) - Q(\mathbf{z} \parallel \mathbf{x}) \right] = 0 \tag{18}$$

for almost every pair $(\mathbf{x}, \mathbf{z})$ relative to the product probability $p(\mathbf{x})q(\mathbf{z})$, for every finite memory l and every finite A.

Proved this point, we can use the parsing of a string $\mathbf{z}$ based on the string $\mathbf{x}$ for estimating the probability that the sequence $\mathbf{z}$ is emitted by p.

For estimating the relative entropy, it is necessary to prove that the quantity $\Delta(\mathbf{z} \parallel \mathbf{x}) = Q(\mathbf{z} \parallel \mathbf{x}) + H_{\mathbf{z}}$ is an estimation of the relative entropy $D(q_{\mathbf{z}} \parallel p)$, i.e.

$$\lim_{n\to\infty} [\Delta(\mathbf{z} \parallel \mathbf{x}) - D(q_{\mathbf{z}} \parallel p)] = 0 \tag{19}$$

Then, the quantity $\Delta(\mathbf{z} \parallel \mathbf{x})$ is an indicator of the relative entropy with the use of the LZ parsing.

The idea to use the parsing of the string $\mathbf{z}$ respect to the string $\mathbf{x}$ for estimating the probability $p_{\mathbf{x}}(\mathbf{z})$ can be understood in the simple case of two bernoullian sequences.

Let $\mathbf{x}$ be a sequence generated by a bernoulli source with probability p for 0 and (1-p) for 1, and let $\mathbf{z}$ be a sequence generated with probability q for 0 and (1-q) for 1, both sequence of length N.

Consider in $\mathbf{x}$ a segment of length n : in this subsequence 0 will appear approximately pn times while 1 will appear approximately $(1-p)n$.

The probability to find a subsequence of this kind in the string $\mathbf{x}$ of length N is:

$$\mathcal{P}_n = p^{np}(1-p)^{n-np} = p^{np}(1-p)^{(1-p)n} \tag{20}$$

The logarithm of this quantity:

$$\begin{aligned} \log_e \mathcal{P}_n &= \log p^{np} + \log(1-p)^{(1-p)n} \\ &= n[p\log p + (1-p)\log(1-p)] \\ &= -nH(p) \end{aligned}$$

and then

$$\mathcal{P}_n = \exp^{-nH(p)} \tag{21}$$

(We remember that this is also the Shannon Mc-Willey theorem [25]) The typical length of a sequence with probability $\mathcal{P}_n$ is $\frac{1}{\mathcal{P}_n}$ [2], and then, the maximal length of the substring that we can find in $\mathbf{x}$ is

$$\frac{1}{\mathcal{P}_n} = \exp^{nH(p)} \sim N \tag{22}$$

we obtain that:

$$nH(p) \approx \log N \tag{23}$$

or rather, the average length of a subsequence is

$$n = \frac{\log N}{H(p)} \tag{24}$$

Remark 2.2.1. *from this average length it is possible to prove also the optimality of the exhaustive parsing, infact:*

$$c(\mathbf{x}) \sim \frac{N}{n} = \frac{N}{\frac{\log N}{H}} = \frac{N}{\log N} H \tag{25}$$

[2] Kac Theorem [60]

and then

$$c(\mathbf{x})\frac{\log N}{N} \sim H \tag{26}$$

Come back to our original sequences $\mathbf{x}$ and $\mathbf{z}$; in $\mathbf{z}$ a subsequence of length n has probability $q^{nq}(1-q)^{n(1-q)}$, where nq and $n-nq = n(1-q)$ is the number of 0 and 1 respectively that we can find in the subsequence of length n. Let w_z be such subsequence. The probability to find w_z in $\mathbf{x}$ must be equal to that of finding a subsequence of length n with the number of 0 and 1 equal to the distribution in the subsequence w_z. Then, the probability of finding the subsequence w_z of length n in $\mathbf{x}$ is given by:

$$\begin{aligned} p^{nq}(1-p)^{n(1-q)} &= \exp^{\log_e(p^{nq}(1-p)^{n(1-q)}} \\ &= \exp^{\log p^{nq}+\log(1-p)^{n(1-q)}} \\ &= \exp^{n(q\log p+(1-q)\log(1-p))} \\ &= \exp^{-n(D(q\|p)+H(q))} \end{aligned}$$

where the last equality follows from.the definition of relative entropy for bernoulian sequences:

$$\begin{aligned} D(q \parallel p) &= q\log\frac{q}{p} + (1-q)\log\frac{(1-q)}{1-p)} \\ &= q\log q + (1-q)\log(1-q) - q\log p - (1-q)\log(1-p) \\ &= -H(q) - q\log p - (1-q)\log(1-p) \\ \Rightarrow q\log p + (1-q)\log(1-p) &= -(D(q \parallel p) + H(q)) \end{aligned}$$

Then we have calculated the probability of a subsequence of length n in $\mathbf{x}$ with the number of zero and ones given by the distribution of the sequence $\mathbf{z}$, or, in other words, the probability that $w_{\mathbf{z}}$ is generated from the source with distribution p.

Following the notations of [115], this quantity is written as $p_{\mathbf{x}}(w_{\mathbf{z}})$.

Even in this case we can calculate the max length of the match subsequence:

$$\frac{1}{p(z)} = \exp^{n(D(q\|p)+H(q))} \sim N \tag{27}$$

$$n(D(q \parallel p) + H(q)) \sim \log N \tag{28}$$

then the max length of the subsequence w_z is equal to: $n_{max} = \frac{\log N}{H(q)+D(q\|p)} = \frac{\log N}{C(q\|p)}$

Similarly to the case of the single entropy:

$$c(\mathbf{z}|\mathbf{x}) \sim \frac{N}{n_{max}} = \frac{N}{\log N}(D(q \parallel p) + H(q)) \tag{29}$$

or, similarly:

$$\frac{\log N}{N} c(\mathbf{z}|\mathbf{x}) \rightarrow_{N\rightarrow\infty} C(q \parallel p) \tag{30}$$

where $C(q \parallel p)$ is equal to $-\log p(\mathbf{z})$ from equation (17)

The statistical properties of the quantity $Q(\mathbf{z} \parallel \mathbf{x})$ are described by the following theorem (Theorem 1 of [115]):

Theorem 2.2.1. *Let $p(.)$ be a stationary Markov source of order l and let μ be an arbitrarily small number. Let $\mathbf{x}$ be a realization of $p(.)$. Then we have the following*

a) *For every fixed $\mathbf{z} \in \mathcal{A}^n$ for which $p(\mathbf{z}) > 0$,*

$$p\left[\mathbf{x} : -\frac{1}{n}\log p(\mathbf{z}) - Q(\mathbf{z} \parallel \mathbf{x}) < 2\mu \log \frac{1}{\delta}\right] \leqslant \exp^{-(\frac{n^\mu}{\log n})K(\delta,\ell)+o(\frac{n^\mu}{\log n})} \tag{31}$$

where

$$\delta = \min_{a^{\ell+1}\in\mathcal{A}^{\ell+1}:p(a^{\ell+1})>0} p(a^{\ell+1}) \tag{32}$$

$$K(\delta,\ell) = \delta^{\ell+1}\log\frac{1}{1-\delta} \tag{33}$$

b) *For every $\mathbf{z} \in \mathcal{A}^n$ for which $p(z) > 0$*

$$p\left[\mathbf{x} : -\frac{1}{n}\log p(\mathbf{z}) - Q(\mathbf{z} \parallel \mathbf{x}) > 2\mu \log \frac{1}{\delta}\right] \leqslant n^{-\frac{\mu}{2}} + 0(n^{-\frac{\mu}{2}}) \tag{34}$$

c) *let $\mathbf{z}$ be a realization of a stationary Markov source of order ℓ' with an underlying probability measure $q(.)$. then (34) can be replaced by a tighter*

bound for almost every $\mathbf{z} \in \mathcal{A}^n$ *(relative to the probability measure* $q(.)$ *) as follows:*

$$p\left[\mathbf{x} : -\frac{1}{n}\log p(\mathbf{z}) - Q(\mathbf{z} \parallel \mathbf{x}) > 2\mu \log\frac{1}{\delta}\right] \leqslant \exp^{-\frac{1}{2}(\frac{\mu^2}{\log^2 n})n^{K^{'}(\delta,\delta^{'})}[1-o(n^{-\mu^2})]} \tag{35}$$

for every $\mathbf{z} \in \mathcal{A}^n - \mathcal{B}$ *for which* $p(z) > 0$*, where* $\mathcal{B}$ *is a subset with the property.*

$$q(\mathbf{z} \in \mathcal{B}) \leqslant \exp^{-K^{''}(\delta,\mathcal{A},\mu)\frac{n^{\mu^{'}}}{\log n}+o(\frac{n^{\mu^{'}}}{\log n})} \tag{36}$$

and

$$\delta^{'} = \min_{a^{\ell+1}\in\mathcal{A}^{\ell+1}:q(a^{\ell+1})>0} q(a^{\ell+1}) \tag{37}$$

$$K^{'} = \frac{1}{4}\frac{\log\frac{1}{1-\delta^{'}}}{\log\frac{1}{\delta}} \tag{38}$$

$$K^{''}(\delta,\delta^{'},\mu) = \frac{\mu}{2(1+\mu)}\frac{\log\frac{1}{1-\delta}}{\log\frac{1}{\delta}}\log\frac{1}{1-\delta^{'}} \tag{39}$$

$$\mu^{'} = 1 - \frac{1}{4}\log\frac{1}{1-\delta^{'}}/\log 1\delta \tag{40}$$

The point a) of the theorem says that $Q(\mathbf{z} \parallel \mathbf{x})$ exceeds $-\frac{1}{n}\log p(\mathbf{z})$ significantly with probability that decays being "almost exponentially". While the point b) says that the $Q(\mathbf{z} \parallel \mathbf{x})$ is significantly smaller than $-\frac{1}{n}\log p(\mathbf{z})$ with probability that decays only polynomially with n. However, from point c), if $\mathbf{z}$ belongs to some subset of high probability under $q(.)$, the latter probability decays "almost exponentially" as well.

The limit (18) is obtained from the point a), b) and the Borell-Cantelli Lemma, while from point a) and b) of the theorem, we have:

$$\lim_{n\to\infty} p\{\mathbf{x} : |-\frac{1}{n}\log p(\mathbf{z}) - Q(\mathbf{z} \parallel \mathbf{x})| > \varepsilon\} = 0$$

2.2.3 The compressor method

The compression algorithm applied to a sequence emitted by an ergodic source $\mathcal{X}$ is able to encode the sequence almost optimally. This optimal

coding will not be the optimal one for another sequence emitted by the ergodic source $\mathcal{Z}$.

The number of bits per character wasted to encode the sequence emitted by $\mathcal{Z}$ with the coding optimal for $\mathcal{X}$ is the relative entropy of $\mathcal{X}$ and $\mathcal{Z}$.

This idea to use the optimal coding for a given source to encode the sequences of another source is fundamental for the distance used in [3].

In order to define the relative entropy between two sources $\mathcal{X}$ and $\mathcal{Z}$ they extract two long sequences $\mathbf{x}$ and $\mathbf{z}$ from the sources $\mathcal{X}$ and $\mathcal{Z}$, respectively, and a short sequence $\overline{\mathbf{z}}$ from the source $\mathcal{Z}$. They append the sequence $\overline{\mathbf{z}}$ to $\mathbf{x}$ and zip this new sequence $\mathbf{x} \wedge \overline{\mathbf{z}}$ with an everyday compressor such as *gzip*. Intuitively, the quantity $\Delta_{\mathbf{x}\overline{\mathbf{z}}} = L_{\mathbf{x}+\overline{\mathbf{z}}} - L_{\mathbf{x}}$ (L_i is the length of the compressed sequence i) measures the length of $\overline{\mathbf{z}}$ in the coding optimized for $\mathbf{x}$, and the relative entropy $\mathcal{S}_{\mathcal{XZ}}$ per character between $\mathcal{X}$ and $\mathcal{Z}$ will be estimated by

$$\mathcal{S}_{\mathcal{XZ}} = \frac{\Delta_{\mathbf{x}\overline{\mathbf{z}}} - \Delta_{\mathbf{z}\overline{\mathbf{z}}}}{|\overline{\mathbf{z}}|} \tag{41}$$

where $\frac{\Delta_{\mathbf{z}\overline{\mathbf{z}}}}{|\overline{\mathbf{z}}|} = \frac{L_{\mathbf{z}\wedge\overline{\mathbf{z}}} - L_{\overline{\mathbf{z}}}}{|\overline{\mathbf{z}}|}$ is an estimation of the entropy of the source $\mathcal{Z}$.

$\mathcal{S}_{\mathcal{XZ}}$ is a *heuristic* compression-based algorithm for $D(\cdot \parallel \cdot)$ and there is no claim that the algorithm converges to $D(\cdot \parallel \cdot)$.

This idea to use compressors based on the LZ77 algorithm (such as *gzip*) for estimating the relative entropy can also be seen as an implementation of the Merhav and Ziv method, where the parsing of the string is replaced by the compression of the string itself. In this analogy, the cross parsing term $c(\mathbf{x}|\overline{\mathbf{z}})$ is given by $\Delta_{\mathbf{x}\overline{\mathbf{z}}}$ while the entropy is estimated by $\Delta_{\mathbf{z}\overline{\mathbf{z}}}$.

Intuitively, the compression algorithm when works out the string $\mathbf{x} \wedge \mathbf{z}$ first of all encodes the sequence $\mathbf{x}$, then it begins encoding the file $\mathbf{z}$.

After having compressed the sequence $\mathbf{x}$, the algorithm starts compressing sequence $\mathbf{z}$ using the dictionary that it has learned from $\mathbf{x}$. After a while, however, the dictionary starts to become adapted to the sequence $\mathbf{z}$, and after a long fraction of $\mathbf{z}$ already analyzed, the dictionary tends to depend only on the specific features of $\mathbf{z}$.

When this appens, it is clear that we are not doing a parsing of the

sequence $\mathbf{z}$ with respect to the only sequence $\mathbf{x}$, then we are not doing the Merhav-Ziv method.

From these considerations it derives the necessity to use the short sequence emitted by the $\mathcal{Z}$ source in the calculation. But, how much "short" must be a $\mathbf{z}$ sequence?

In [94] the authors analyze accurately this problem and show that it exists a *crossover length* for the sequence $\mathbf{z}$, which depends on the relative entropy between $\mathbf{x}$ and $\mathbf{z}$, below which the compression algorithm does not learn the sequence $\mathbf{z}$ (measuring in this way the cross entropy between $\mathbf{x}$ and $\mathbf{z}$) and above which it starts learning $\mathbf{z}$, i.e. optimizing the compression using the specific features of $\mathbf{z}$.

Then, when we use the adaptive compressors such as those based on the LZ parsing, the implementation of the Merhav and Ziv method needs some care. To implement exactly the Merhav and Ziv method with the compressors implies the necessity to study this crossover length for the sequence $\mathbf{z}$ (appended to $\mathbf{x}$).

Following the same discussion we made in Merhav and Ziv theorem for two bernoullian sequences, a string of length n_z in $\mathbf{z}$ has a probability of occurrence equal to $e^{-n_z h_{\mathbf{z}}}$, while the probability of occurrence of the same string in the string $\mathbf{x}$ is equals to the probability of the string respect to the measure p and is asymptotically given by $e^{-n_z[h_{\mathbf{z}}+D(q\|p)]}$. [3]

As already discussed, the typical distance between two occurrences of the same substring is inversely proportional to the probability of the substring itself.

Then, if we look for the length n_z of the longest match found in $\mathbf{z}$:

$$L_z \sim \frac{1}{\text{probability of a typical sequence in } \mathbf{z} \text{ of length } n_z} = e^{n_z h_z}$$

from which we obtain the value of

$$n_z = \frac{\log L_z}{h_{\mathbf{z}}}$$

[3] this results is based on the Shannon-McMillan theorem plus the theory of types [25] pag281

Otherwise, if we want the length n_x of the longest subsequence of $\mathbf{z}$ in $\mathbf{x}$

$$L_x \sim \frac{1}{e^{-n_x[h_{\mathbf{z}}+D(q\|p)]}} = e^{n_x[h_{\mathbf{z}}+D(q\|p)]}$$

from which we get:

$$n_x \sim \frac{\log L_x}{h_{\mathbf{z}} + D(q \parallel p)}$$

Now, intituively, if $n_z \ll n_x$ then the longest match will be found in $\mathbf{x}$ rather than in $\mathbf{z}$.

Rewriting the condition $n_z \ll n_x$ respect to the lengths of the two sequences, we obtain:

$$\frac{\log L_z}{\log L_x} \ll \frac{h_{\mathbf{z}}}{h_z + D(q \parallel p)} \tag{42}$$

According this condition, we can use the compressor algorithms for implemeting the idea of the Merhav and Ziv (because we expect that the longest match will be found in $\mathbf{x}$), and so, we can use the compressor algorithms for estimate the relative entropy between two different sources.

Moreover, in [94] there is also a conjecture about a scaling function which rules the way in which the compression algorithms learn the sequence $\mathbf{z}$ after having zipped the sequence $\mathbf{x}$; this scaling function is connected to an analysis of the fluctuations on the transient region (i.e. the region where the sequence $\mathbf{z}$ starts and the compressor begins a sort of learning process).

The conjecture says that, when the lengths of the two sequences $\mathbf{x}$ and $\mathbf{z}$ go to infinity, the probability of the learning function $P(\mathbf{x}, \mathbf{z})$ (i.e. the probability that, once the zipper starts scanning the $\mathbf{z}$ part of the $\mathbf{x}$ +$\mathbf{z}$ file, it finds a matching in the $\mathbf{x}$ part rather than in the $\mathbf{z}$ part) converges to a unique function:

$$P(\mathbf{x}, \mathbf{z}) \rightarrow_{\mathbf{x},\mathbf{z}\rightarrow\infty} f\left(\frac{\mathbf{x} - \alpha\mathbf{z}}{\sqrt{\mathbf{x} + \mathbf{z}}}\right)$$

where $\alpha = \frac{h_{\mathbf{z}}}{h_{\mathbf{z}}+D(q\|p)}$ and $\sqrt{\mathbf{x} + \mathbf{z}}$ is the random fluctuations around its average.

Otherwise, it is not evident (and perhaps neither true) that with the compression of two appended sequences $\mathbf{x}$ +$\mathbf{z}$ minus the compression of the file $\mathbf{x}$ we are estimating the cross entropy of two files.

2.2.4 The compression of two appended sequences

The difficulty to understand the teoretical quantities involved in the compression of two appended files is due to the use of adaptive LZ compressors.

These algorithms belong to the class of Universal Code, that works without *a priori* knowledge of the statistics of the sequences, converging anyway, and quickly to the sequence entropy itself.

In the case of statistical compressor algorithms, it is possible to detect the quantities involved in the compression of two appended files.

We remember that the algorithms based on statistic coding work in two steps: first it is constructed the histogram representing the frequency of each byte and is constructed an optimal code so that the probability of a symbol has a direct bearing on the length of its representation (a sequence is modeled as an i.i.d. Bernoulli process). In the second step, the algorithm reads again the file and encodes it using the code it has constructed.

Consider two files $\mathbf{x}$ and $\mathbf{y}$ with empirical distribution $p_{\mathbf{x}}$ and $p_{\mathbf{y}}$ given by:

$$p_{\mathbf{x}}(i) = \frac{\overline{N}_{\mathbf{x}}(i)}{L_{\mathbf{x}}} \; ; \; p_{\mathbf{y}}(i) = \frac{\overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{y}}} \tag{43}$$

where with $\overline{N}_{\mathbf{x}}(i)$ we indicate the occurrence of i in $\mathbf{x}$, and $L_{\mathbf{x}}$ $L_{\mathbf{y}}$ are the length of $\mathbf{x}$ and $\mathbf{y}$ respectively.

Then $\mathbf{z} = \mathbf{x} \wedge \mathbf{y}$ has empirical distribution equal to:

$$p_{\mathbf{z}}(i) = \frac{\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{x}} + L_{\mathbf{y}}} \tag{44}$$

The number of bits needed to encode the file $\mathbf{z}$ with a static coder, for example an arithmetic coder C, is the size of the built histogram ($|hist|$) and

the entropy of $p_{\mathbf{z}}$ times the number of simbols:

$$\begin{aligned} C(\mathbf{z}) &= |hist| + (L_{\mathbf{x}} + L_{\mathbf{y}})H(p_{\mathbf{z}}) \\ &= |hist| + (L_{\mathbf{x}} + L_{\mathbf{y}}) \sum_{i \in \mathcal{A}} -p_{\mathbf{z}}(i) \log p_{\mathbf{z}}(i) \\ &= |hist| - (L_{\mathbf{x}} + L_{\mathbf{y}}) \sum_{i \in \mathcal{A}} \frac{\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{x}} + L_{\mathbf{y}}} \log \frac{\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{x}} + L_{\mathbf{y}}} \\ &= |hist| - \sum_{i \in \mathcal{A}} (\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)) \log \frac{\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{x}} + L_{\mathbf{y}}} \end{aligned} \tag{45}$$

This quantity can be rewritten as following (we omitted the term about the histogram frequencies):

$$C(\mathbf{z}) = -\sum_{i \in \mathcal{A}} L_{\mathbf{x}} \frac{\overline{N}_{\mathbf{x}}(i)}{L_{\mathbf{x}}} \log \frac{\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{x}} + L_{\mathbf{y}}} + L_{\mathbf{y}} \frac{\overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{y}}} \log \frac{\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{x}} + L_{\mathbf{y}}}$$

Summing and subtracting the quantities $\sum_{i \in \mathcal{A}} \frac{\overline{N}_{\mathbf{x}}(i)}{L_{\mathbf{x}}} \log \frac{\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{x}} + L_{\mathbf{y}}}$ and $\sum_{i \in \mathcal{A}} \frac{\overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{y}}} \log \frac{\overline{N}_{\mathbf{x}}(i) + \overline{N}_{\mathbf{y}}(i)}{L_{\mathbf{x}} + L_{\mathbf{y}}}$ we obtain:

$$C(\mathbf{z}) = L_{\mathbf{x}} H(p_{\mathbf{x}}) + L_{\mathbf{y}} H(p_{\mathbf{y}}) + L_{\mathbf{x}} D(p_{\mathbf{x}} \parallel p_{\mathbf{z}}) + L_{\mathbf{y}} D(p_{\mathbf{y}} \parallel p_{\mathbf{z}}) \tag{46}$$

So, in this case, the compression of two appended files calculates the entropy of the single files plus the total Kullback-Leibler Divergence to the "mean" distribution $p_{\mathbf{z}}$.

The quantity $D(p_{\mathbf{x}} \parallel p_{\mathbf{z}}) + D(p_{\mathbf{y}} \parallel p_{\mathbf{z}})$ is also named as the Jensen divergence and it is equal to $H(p_{\mathbf{x}} + q_{\mathbf{z}}) - H(p_{\mathbf{x}}) - H(q_{\mathbf{z}})$.

On the other hand, with real data, not generated as any finite order chain and modern compressors based on the lempel ziv procedure, it is not clear whether an analogue of (46) still holds. Some conjectures [62], and numerical results [94] seem to indicate that the compression of two concatenated sequences can be related to the relative entropy; nevertheless, as seen in the previous section, the use of compressor algorithms for estimating the relative entropy is justified only when we "force" the compression of $\mathbf{z}$ on $\mathbf{x}$ (when we try to implement the Merhav and Ziv theorem) otherwise, we don't have

theorems or results showing us where the compression of two appended files go, when $n \to \infty$. Moreover, also in [24], the construction of the NCD distance from the NID distance, suggests the use of the compression of two sequences correlated to an estimation of the joint entropy, but also this fact is not evident in the case of real adattive compressor algorithms.

2.2.5 BWT

The distance that now we show is proved to be a good estimator of the relative entropy. The core of this distance is a transformation on strings (BWT).

The Burrows Wheeler Transform (BWT) [8] acts as a permutation on any given finite string and this reversibile sequence transformation is used in data compression techniques such as bzip2.

In the case of a sequence generated by an unknown markov source, the BWT tends to group together characters sharing the same state: in particular, the BWT converts any given markovian sequence into a piecewise i.i.d. memoryless sequence. In this way we can obtain a string $BWT(w)$ that can be compressed in a more efficient way through traditional statistical compressors.

Briefly, these are the steps of the algorithm:

- first it is necessary to construct a matrix M which consists of all cyclic shifts of w; for example: If $w = abraca$ then the matrix M will be:

  a b r a c a
  b r a c a a
  r a c a a b
  a c a a b r
  c a a b r a
  a a b r a c

- then the rows of the matrix M are sorted lexicographically keeping track of the position I of the original string w in the new matrix:

$$
\begin{array}{cccc}
 & & 0 & \mathrm{a\,a\,b\,r\,a\,c} \\
\mathrm{I} & \rightarrow & 1 & \mathrm{a\,b\,r\,a\,c\,a} \\
 & & 2 & \mathrm{a\,c\,a\,a\,b\,r} \\
 & & 3 & \mathrm{b\,r\,a\,c\,a\,a} \\
 & & 4 & \mathrm{c\,a\,a\,b\,r\,a} \\
 & & 5 & \mathrm{r\,a\,c\,a\,a\,b}
\end{array}
$$

- the last column of the matrix obtained represents $BWT(w)$ (in the example $BWT(w) = caraab$ and $I = 1$ since the original sequence appears in row 1)

The Burrows-Wheeler is a reversibile transformation, indeed. Given $BWT(w)$ and the index I, it is possible to recover the original string w. For the interesting asymptotic properties and combinatorial results on this transformation refer to [8, 9].

Few years ago it has been realized that this transformation is not only useful for data compression, but that it can be turned into a very efficient entropy estimator for any given unknown Markov source [31] as well.

The proof of this convergence is very rigously in [31] and [10]. Here, using the definition of *empirical entropy* given in [68], we want to show how this transformation allows the entropy estimation using the definition of *empirical entropy* given in [68].

Let s be a string of length N over the alphabet $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_h\}$, and let n_i denote the number of occurrences of the symbol α_i inside s.

The zeroth order entropy is defined as

$$H_0(s) = -\sum_{i=1}^{h} \frac{n_i}{N} \log \frac{n_i}{N} \tag{47}$$

where, as usually, we assume $0 \log 0 = 0$.

For any string w and $\alpha_i \in \mathcal{A}$, let $n_{w\alpha_i}$ denote the number of occurrences in s of the string w when followed by α_i. Let $n_w = \sum_{i=1}^{h} n_{w\alpha_i}$. Then, the kth

order entropy is defined as:

$$H_k(s) = -\frac{1}{|s|} \sum_{w \in \mathcal{A}^k} \left(\sum_{i=1}^{h} n_{w\alpha_i} \log \frac{n_{w\alpha_i}}{n_w} \right) \tag{48}$$

It is possible to rewrite this formula in the following way [79]: for any length-k word $w \in \mathcal{A}^k$ let w_s denote the string consisting of the characters following w inside s, or otherwise, the string of the characters with the same context.

The length of w_s is equal to the number of occurrences of w in s.

Then, the zero order empirical entropy for the string w_s is $H_0(w_s) = -\sum_{i=1}^{h} \frac{n_{is}}{|w_s|} \log \frac{n_{is}}{|w_s|}$ where with n_{is} we denote the frequency of the α_i character in the string w_s. But the frequency of the character α_i in w_s is equal (according the construction rule of w_s) to the frequency of the character α_i with the previuos context equals to w, denoted with $n_{w\alpha_i}$. Then: $H_0(w_s) = -\sum_{i=1}^{h} \frac{n_{w\alpha_i}}{|w_s|} \log \frac{n_{w\alpha_i}}{|w_s|}$ that, substituted in (48) and multiplying for $|w_s|$ we obtain the following formulation of the $k-$order empirical entropy:

$$H_k(s) = -\frac{1}{|s|} \sum_{w \in \mathcal{A}^k} |w_s| H_0(w_s) \tag{49}$$

In this last formulation of the empirical entropy is more evident the connection with the BWT. If we take the string s and we apply the BWT transformation, the $BWT(s)$ has the following remarkable property, indeed: for each substring w in s, the characters following w in s are grouped together inside $BWT(s)$. Then, if we cut the $BWT(s)$ where the context changes, each result of this cuts is a substring such as the previous w_s.

Then, the BWT transforms the string s in another string $BWT(s)$ for which the computation of the $k-$order empirical entropy is more easily.

An important results in [10] proves that the BWT output sequence is close to a piecewise stationary memoryless source, then we can segment the output sequence and estimate the probabilities in each segment.

However, these transition points depend on the unknown memory structure of the underlying source. How to segment the string then? A natural

approach is an adaptive segmentation, that tries to base the segmentation on the empirical distribution of the BWT output in order to detect the transitions.

However, from experimental results, always in [10] is showed that a simple Uniform segmentation method performs almost as well as the more complex adaptive method in most case.

Very recently Cai at al. in [9] developed these ideas and constructed an efficient relative entropy (Kullback-Leibler Divergence) indicator.

The core of this task is always the approximation of the quantity $\log_2 p_x(z^n)$, as already seen in the Merhav and Ziv section.

In [9] they run the BWT on the reversed concatenation of files $\mathbf{x}$ and $\mathbf{z}$, and then partition the $BWT(\mathbf{x} \wedge \mathbf{z})$ according to the symbols in $\mathbf{x}$. Usually a uniform segmentation strategy is chosen, i.e. the $BWT(\mathbf{x} \wedge \mathbf{z})$ is divided into segments so that each segment contains an equal number of symbols from the sequence $\mathbf{x}$. Clearly, the segments of the $BWT(\mathbf{x} \wedge \mathbf{z})$ that contain both symbols belonging to $\mathbf{x}$ and $\mathbf{z}$ may be different in length because the numbers of symbols from $\mathbf{z}$ can be different.

Given the segmentation of the $BWT(\mathbf{x} \wedge \mathbf{z})$ according to $\mathbf{x}$ one estimates the first order distribution (with respect to $\mathbf{x}$) within each segment j:

$$\overline{p}_{\mathbf{x}}(A, j) = \frac{\overline{N}_j(A) + \Delta}{\sum_{B \in \chi} \overline{N}_j(B) + |\chi|\Delta} \tag{50}$$

where A is a character from an alphabet χ, $\overline{N}_j(A)$ denotes the number of occurrences of symbol A from $\mathbf{x}$ in the jth segment and Δ is just a suitable bias (due to the uniform segmentation choice).

The contribution of the jth segment to the approximation of $\log_2 p_x(z^n)$ is: $\log_2 p_x(j) = \sum_{a \in \chi} \overline{N}_j(a) \log_2 \overline{p}_{\mathbf{x}}(A, j)$ where with the lower case notation we indicate characters from the sequence $\mathbf{z}$. Finally, the approximation of our initial cross term is given by an average across the T_x segments:

$$\overline{C}(q_z \| p_x) = -\frac{1}{n} \sum_{j=1}^{T_x} \log_2 \overline{p}_{\mathbf{x}}(j) \tag{51}$$

where $\overline{C}(q_z \| p_x)$ is the usually cross entropy also defined in equation (16).

Even if the uniform segmentation is not, in general, the optimal one (in principle one should cut the sequences at the (unknown) state transition point), in the case of uniform segments of length of order $\approx |\mathbf{x}|^{\frac{1}{2}}$ one can prove a convergence (in probability) of the estimator for sequences emitted by finite-alphabet finite-state Markov sources [9, 31]; moreover, the rate of convergence appears quite fast in respect of other empirical relative entropy indicators.

For estimating the term $H(q_z)$ it is then enough to repeat the similar segmentation procedure on the $BWT(\mathbf{z})$.

In this way, we have an estimation of the relative entropy via Burrows Wheeler Transform:

$$\overline{D}(q_z \| p_x) = \overline{C}(q_z \| p_x) - H(q_z) \tag{52}$$

2.2.6 The n-gram distance

The distance shown now is based on the computing of the $n$gram probability distribution.

Given a sequence S with each character belonging to an alphabet $\mathcal{A}$, a *n-gram* of the sequence is a subsequence of length n. The i^{th} n-gram of S is the sequence $(s_i, s_{i+1}, \ldots, s_{i+n-1})$.

In a string of length N there are $N - n - 1$ n-grams; the distinct n-grams found the *Dictionary* D_S of the sequence. The max cardinality of the dictionary can be equal to $\sharp(\mathcal{A})^n$, if $\sharp(\mathcal{A})^n < |S|$ (where $|S|$ is the length of S).

Given a sequence, $S = (s_1, s_2, \ldots, s_N)$, for fixed n, we can calculate the frequency $f_S(w_i)$ of n-grams $w_i = (s_i, s_{i+1}, \ldots, s_{i+n-1})$ that appears in the sequence S $(i = 1, \ldots, N - n - 1)$.

In [65] the authors, following the approach of Bennet in [5] with the bi-grams, use the n-grams frequencies in sequences for constructing a similarity

distance between sequences:

$$d_n(S_1, S_2) = \sum_{w \in D_{S_1} \cup D_{S_2}} \left(\frac{f_{S_1}(w) - f_{S_2}(w)}{\frac{f_{S_1}(w) + f_{S_2}(w)}{2}} \right)^2$$

where D_{S_k} is the set of all possible words of length n in the string S_k (the *dictionary*), and $f_{S_k}(w) = 0$ if $w \notin D_{S_k}$. This distance has been used in the authorship attribution in literary texts with quite positive results. We refer to [65] for the details.

In our experiments, we use a suitable normalized version of the above formula as a useful indicator of the similarity between two sequences:

$$d_n^{\alpha}(S_1, S_2) = \frac{1}{(\sharp D_{S_1})^{\alpha-1} + (\sharp D_{S_2})^{\alpha-1}} \times \sum_{w \in D_{S_1} \cup D_{S_2}} (\frac{(f_{S_1}(w) - f_{S_2}(w))^2}{(f_{S_1}(w) + f_{S_2}(w))^{\alpha}} \quad (53)$$

In the applications presented in the next chapters we also make use of another (pseudo) distance, again introduced by Keselj in studing DNA sequences [105]. More precisely, given the frequencies vectors of the n-grams in two symbolic strings, we define the "*geometric formula*" as:

$$d_n^{geo}(S_1, S_2) = \frac{1}{2} \sum_{w \in D_{S_1} \cup D_{S_2}} \frac{\mid f_{S_1}(w) - f_{S_2}(w) \mid}{\sqrt{f_{S_1}(w) f_{S_2}(w)} + 1} \quad (54)$$

Chapter 3

Heartbeat Signals

In this chapter we introduce and describe a recent approach to heuristic estimation of similarity between symbolic sequences.
While these methods have been already applied to various classes of sequences (such as DNA sequences and literary texts), we want here to suggest the possibility of using this approach also for information extraction and classification of heart rate variability (HRV) sequences.
Most of the results showed in this chapter are published in [27], [28].

3.1 The time serie ECG

ECG signal is a paradigmatic example of non-linear, non-stationary noisy process. This 1-dimensional time series reflects the net results of an enormous number of interactions between the cardiovascular system, the autonomous nervous system and the external environment; nevertheless they still contains *valuable information* concerning the clinical/pathological state of the *source* [47].

Various and quite sophisticated techniques are presently available for *extracting useful information* out of the ECG signal. These approaches range from non-linear methods developed in the theory of finite dimensional dynamical systems to time-domain and frequency-domain spectral analysis [103], as the recent stimulating paper [80] shows. Interesting tools out of linguistic analysis have recently been used to study human heartbeat [111].

Heart rate, defined in terms of the number of myocardial contractions, is

a complex entity and lays under a plethora of regulatory factors (i.e.: autonomic nervous system, endocrine setting, circuitry resistance, cell membrane plasticity, etc.[36, 12, 33, 56]) some of which acts even during chaotic functional states like fibrillation [108].

Althought physiology and medicine have investigated for years the dynamics behind its functionality and "behavior", it still remains largely unknown the weight to assign to each single factor (these weights may probably change too [112]) or to what extent stochastic phenomena may account for them. With such premises it is interesting to discuss the nature of a widely known investigation method: Heart Rate Variability (HRV), defined as the difference, in time, between two following heart beat lags. HRV accounts for a large portion of the homeostatic efforts of the individual, it is an essential part of stress: it is quickly changing to grant adaptation to every life-compatible circumstances/stimuli, and furthermore it also shows long-range correlations [63].

This is an interesting way to investigate and understand HRV. Nevertheless while HRV analysis, introduced in clinical practice in the late 'sixties in obstetrics, has been used in many studies to investigate the most different functional parameters in human beings (even the effects of geomagnetism on health [89]) and has yield to the development of an incredible huge number of analytical methods [47] dedicated to its investigation, little light has been shed on its foundation. HRV has proved to be a fantastic tool to evaluate autonomic system function [40], in particular when it is studied through quite elementary use of the Fourier Analysis [47], and it has proven to be an independent gauge of life expectancy [95], when undergone an even simpler elaboration (i.e. statistical evaluation of the time series distribution)[47]. Unfortunately when it comes to investigate deeper properties of an individual, the real effectiveness of HRV is still unclear, and this posed a challenge to many researchers to built more and more complex and efficient methods for extracting *valuable* information out of HRV data [61, 81].

3.1.1 Symbolic ECG analysis: RR, HRV and HkV

In our experiments we considered 24 h. Holter ECG signals, obtained from [43] and also from [80].

For applying our methods on these kind of signals we need to code an ECG time series in a simbolic sequence.

The necessary coding procedure is the following: first of all we extract RR interval sequence (R_j) from the full ECG signals, losing in particular *all* the information contained between these two events, such as the P wave, QRS complex, ST segment, ecc...Then we perform an elementary binary coding by looking at the sign of variability, i.e. we construct a new $0, 1$ sequence by setting $w_j = 0$ if $R_j - R_{j-1} > 0$, and $w_j = 1$ otherwise. We obtain the so-called HRV binary sequences representing the sign of the (discrete) first derivative of the RR interbeat time series.

It is important to remark that after this process the original information has been hugely reduced: an original 24 hours ECG signal of few Mega byte (Mb) is reduce to a binary file of about 100 Kb.

In our opinion it is remarkable that even after this tremendous and elementary reduction of the signals, our distance is still capable of capturing *common features* in the signals as our positive and consistent results in all the classification experiments shown.

In other words, this binary coding based on the decreasing or increasing of two consecutive RR intervals is the most elementary one.

More refined coding, for example based on the magnitudes of variability explored in the PNNx statistics [84] are currently under investigation. A small increasing of the vocabulary used in coding of the RR sequences should bring to a better performance, while still keeping the computational complexity into practical affordable limits.

We explored also different ways to encode ECG signals, for testing whether the choice of the symbolic coding HRV extracted from the full ECG is really optimal or - on the contrary - whether the use of some coarse graining on the plain RR interbeat sequence may be sufficient to some classification

purposes. We also explore the possibility of using longer time correlations of the RR signal, called HkV series, based on the observation and the coding of the signal variability over a wider time-window.

Every full ECG signal was preprocessed in order to extract the RR normal interbeat sequence $(R_j)_{j\geqslant 1}$.

First, a symbolic string over a finite alphabet was associated to each RR signal, as follows. Given some graining size $2 \leqslant g \leqslant 200$, we considered a uniform partition of the interval $[R_{min}, R_{max}]$. In the symbolic string $RR[g]$ -representing the numerical series RR at graining g- there is symbol $k \in \{1, 2, \ldots, g\}$ at site j if R_j belongs to the $k-$th interval of the uniform partition.

Subsequently, we extracted the HRV coding of the RR series by looking at the sign of variability, i.e. we built a new binary sequence (the HRV series) by setting $w_j = 0$ if $R_j - R_{j-1} > 0$ and $w_j = 1$ otherwise.

This binary coding can be generalized to *higher order* by defining the corresponding natural coding that gather information about k consecutive RR intervals [14, 13]. This coding will be denoted by HkV coding and it is defined as follows. Let $k \geqslant 1$ be a positive natural number, and fix a labelling of the space of permutation on k object $\mathcal{S}_k$ through a given alphabet $\mathcal{A}$ of size $|\mathcal{S}_k| = k!$. Given any arbitrary RR sequence $(R_1, R_2, R_3, \ldots, R_n)$ and any $1 \leqslant j \leqslant n - k$, we let $w_j \in \mathcal{A}$ corresponding to the unique partition $\pi \in S_k$ that orders the k consecutive RR intervals $(R_{j+1}, R_{j+2}, \ldots, R_{j+k})$. Notice that for $k = 1$, the new sequence $(\omega_1, \omega_2, \ldots)$ coincide with the previously defined binary HRV coding.

3.1.2 Entropy saturation

For each graining size g, we calculated the exhaustive parsing of $RR[g]$ and its corresponding complexity $c(RR[g])$. As already stated, let us define the corresponding entropy by $h_E(RR[g]) = c(RR[g])\frac{\log_2(L)}{L}$, where L is the length of the RR series. Note that for *i.i.d.* R_j's uniformly distributed in $[0, 1]$, we would have $h_E \approx \log g$, and we will use this normalization later on.

Numerical experiments on our set of data show the following *saturation effect*: the entropy $h_E(RR[g])$ increases in g logarithmically up to some g^*. For g exceeding this critical value g^*, the entropy h_E remains constant and that saturation value may be read as the entropy h of the RR series. Figure 3.1 shows two examples of saturation for patients belonging to the **gk-nk** dataset (see next section 3.2.1).

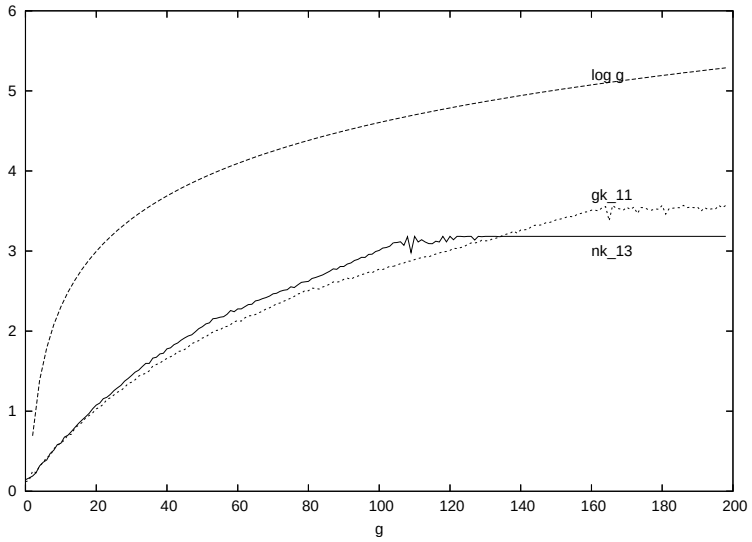


Figure 3.1 ◇ Two examples of saturation $h_E(RR[g])$: an healthy patient (gk_11) and a hospitalized patient (nk_13). The plots are compared with the i.i.d. behaviour $\log(g)$. The two patients belong to the **gk-nk** dataset (see later)

3.2 Clustering RR and HRV via entropy

In this section we evaluate if the singol value of complexity for a sequence is able to discriminate between different groups. The dataset used in this section is the **gk-nk** dataset which description is given in the section 3.2.1.

The saturation value h of RR sequences depends on the patient such as the critical value g^*, which always lies between 100 and 200. To obtain com-

parable symbolic sequences, we chose to fix the graining to $g = 200$. Hereafter, when we refer to RR sequences, it will mean that we are considering numerical series RR at graining size equal to 200 symbols, with the entropy normalized by a factor $\log_2(200)$. Thus, for each patient from the two data sets, we have analysed the two sequences RR[200] and HRV and computed the corresponding entropies $h \in [0, 1]$: $h = c(x)\frac{\log_2 n}{n \log_2(200)}$ for RR[200], while for HRV $h = c(x)\frac{\log_2 n}{n}$.

This experiment has two main aims. First, comparing the amount of information that can be achieved from either RR or HRV concerning the physical status of the patient. Second, verifying whether individual entropies could be sufficient to discriminate among the different classes of ECG signals.

We ordered both families **gk** and **nk** of files with respect to the increasing values of the entropy. The following points are straightforward and true for the complete database. For the sake of simplicity, we shall include tables and plots only for 20 subjects (10 **gk** and 10 **nk**), randomly chosen. Table 3.1 shows entropy values for that sample subset of data.

(i) As we already know from standard statistical and nonlinear methods applied on HRV data, variability is decreased in the case of heart failure. This feature is confirmed by the entropy calculation even in the case of plain RR analysis. In fact, let $\sigma_{nk}(RR)$ be the mean values of the entropy of RR[200] over all the hospitalized patients and $\sigma_{gk}(RR)$ be the mean value over the complete set of healthy patients. Let $\sigma_{nk}(HRV)$ and $\sigma_{nk}(HRV)$ be defined similarly for HRV signals. We have that

$$\sigma_{nk}(RR) = 0.46 \quad < \quad \sigma_{gk}(RR) = 0.51$$

and

$$\sigma_{nk}(HRV) = 0.82 \quad < \quad \sigma_{gk}(HRV) = 0.87$$

The entropy values of both RR and HRV sequences corresponding to **gk** subjects are on average higher than the corresponding sequences coming from **nk** subjects.

(ii) After ordering the RR files it is not useful to distinguish the two classes. The sets are definitely mixed up and a borderline can not be identified. Figure 3.2 (bottom graph) shows what happens for the sample subset of patients.

(iii) The individual HRV sequences allow some kind of discrimination between classes, even if a certain number of mismatches occurs (see an example on Figure 3.2, top graph). This performance may be improved by taking into consideration pairwise differences between sequences and discussing the classification reached by means of the similarity distance $d(\cdot,\cdot)$ induced by the exhaustive parsing, as we will later discuss.

(iv) even if the individual entropies deviate a little bit less around their mean value, these pointwise quantities are still *not sufficient* to achieve a good classification performance also for higher k's (usually, in the experiments we have performed on our data with $k = 1, 2, 3, 4$)

3.2.1 The datasets description

We resume here the description of the principal datasets used in our experiments. We will refer to the dataset used from time to time, according to the different tasks faced.

The data used come from two main sources: the Physionet archive [43] and data of [80] and available upon request.

1. **Healthy Unhealthy dataset:**

   (a) **gk-nk dataset:**

   As described in [80] and repeated here, two main groups of patients have been used:

   nk group made of 90 patients hospitalized during 2001-2004 in the 1st Department of Cardiology of Medical University in Gdańsk, Poland (9 women, 81 men, the average age is 57 $\pm$10)

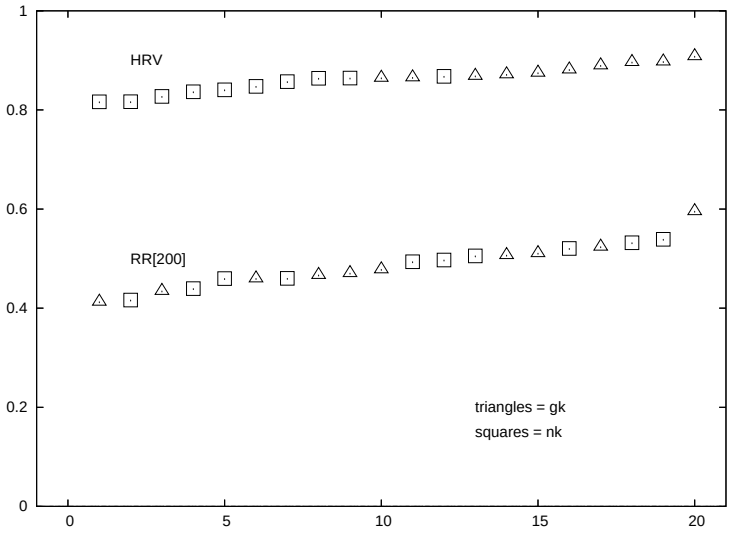


Figure 3.2 ⋄ A plot of the classification via entropy h for a sample subset of 20 subjects. The results referred to healthy patients are plotted by triangles, the ones to hospitalized patients by squares. HRV signals brought to two classification errors (top graph), while the same analysis on the RR[200] signals is not able to discriminate at all (bottom graph).

in whom the reduced left ventricular systolic function was recognized by echocardiogram due to the low left ventricular ejection fraction (LVEF $\leqslant 40\%$, mean LVEF $= 30,2 \pm 6,7\%$);

gk group made of 40 healthy individuals (4 women, 36 men, the average age is 52 ± 8) without past history of cardiovascular disease, with both echocardiogram and electrocardiogram in normal range.The left ventricle ejection fraction was normal (mean LVEF $= 68,0 \pm 4,7\%$).

(b) **nsr-chf dataset:** These data belong to the Physionet archive [43].

chf group (Congestive Heart Failure) This database includes beat annotation files for 29 long-term ECG recordings of subjects aged 34 to 79, with congestive heart failure (NYHA classes I,

II, and III), plus other 15 subjects (11 men, aged 22 to 71, and 4 women, aged 54 to 63) with severe congestive heart failure (NYHA class 3-4)

nsr group (Normal Sinus Rhythm) This database includes beat annotation files for 30 long-term ECG recordings of subjects in normal sinus rhythm (30 men, aged 28.5 to 76, and 24 women, aged 58 to 73) plus 18 long-term ECG recordings of 5 men, aged 26 to 45, and 13 women, aged 20 to 50.

Both original ECG recordings were digitized at 128 samples per second, and the beat annotations were obtained by automated analysis with manual review and correction.(i 15 della classe III e IV 250 samples per second...)

(c) **nsr_w-chf_w dataset:** In our experiment we will use also only the 5 hours portion of the signals that correspond in general to the patients being awake (denoted by **nsr_w** and **chf_w**). These last kind of signals come from [80].

nsr_wake group : these are 13 healthy subjects belonging to [52] from which the wake parts of the signals have been extracted

chf_wake group : wake part of the signals corresponding to 13 subjects with congestive heart failure

2. **young-old dataset:**

(a) **old_gk-young_nsr dataset:** From the previously datasets of healthy patients, we have extracted a subset:

old group 13 healthy subject belonging to **gk** previously described.

young group 13 healthy and rather young people (age between 20-40 years). These patients (3 men, 10 women) show no significant arrhythmias. The corresponding ECG recordings

are available from the Physionet archive [52] [1]

(b) **nsr_old-nsr_young dataset:** Another dataset of young and old patients is extracted from the dataset [53]:

nsr_old: numero healthy individuals of age from 60 to 68 years

nsr_young: numero healthy individuals of age from 28 to 40 years

(c) **fantasia dataset:** Moreover, we have used data from physionet [43]:

f_y Twenty young (21 - 34 years old)

f_o twenty elderly (68 - 85 years old)

both rigorously-screened healthy subjects underwent 120 minutes of continuous supine resting while continuous electrocardiographic (ECG), and respiration signals were collected; in half of each group, the recordings also include an uncalibrated continuous non-invasive blood pressure signal(f2). Each subgroup of subjects includes equal numbers of men and women.

3. **NYHA classification:** class of individual patients with classified congestive heart failure. Various Holter ECG's files were downloaded from [51] and consists of patients belonging to class I,II and III of the NYHA classification.

3.2.2 The attribution methods

The application of a distance on a dataset gives a distance matrix with all the values of the distances between the sequences. For each distance formula we construct a distance matrix of dimension $N \times N$ where N is the number of the sequences under analysis. In each row, then, it is stored the value of the distance of a sequence X_i from all other sequences X_j, with $i, j = 1, \ldots, N$.

[1] The original physionet database consists of 18 nsr records, from these we have removed patients with age greater than 40 years

Given the distance matrix, we used different methods to assign a sequence to a specific class:

- *Mean:* a given test sequence t is attributed to the class with minimal average distance

- $k-$*Nearest Neighbor Method* (here with $k = 1$): a sequence is classified by a majority vote of its neighbors, with the object being assigned the class most common among its k nearest neighbors.

- *vertical voting*: this is a kind of weighted average: taken a test sequence t, all other training sequences belonging to the two classes are sorted according the value distance from the test file t. The vote is calculated only on the files position; in the case of two classes, class A and class B, for example, we calculate: $v_A(t) = \sum_{i=1}^{n_A} \frac{k_j}{j}$ where, k_j is the position of $j-$th file of class A in the sorted list and n_A is the number of the training sequences for the class A. Similarly we can define the vote $v_B(t)$ for the class B and if $v(t) = \frac{v_A(t)-v_B(t)}{v_A(t)-v_B(t)}$ is positive then the sequence t is attributed to the class A otherwise to the other class.

We finally want to stress that in all attribution methods we have examined the prediction quality in two ways: the first one is based upon resubstitution test and the other upon the jacknife test. The use of these methods is on one side for testing the self consistency of the distance, and on other for testing the results by cross-validation.

3.3 Clustering RR and HRV via exhaustive distance

As the previous analysis shows, the individual complexity and entropy of the sequences are not enough to discriminate in a satisfactory way the healthy from the hospitalized patients. We then turn to consider a method of clus-

tering based on the exhaustive distance, or *LZ-distance* $d_{LZ}(\cdot,\cdot)$ defined in section 2.1.3.

3.3.1 The gk-nk dataset

In this section we describe the numerical experiments performed with our method on several ECG's from different groups of subjects, where the ECG's signals have been coded as previously described.

Initially, we have analyzed and used the data obtained from the group of research from Gdańsk University (Poland) (the **gk-nk** dataset), with the aim of comparing our results with the ones obtained by them using multifractal analysis [80]. In particular, they performed classification tasks by calculating the local exponents of the spectrum associated to the RR series with the use of the Wavelet Transform Modulus Maxima Method and also with the use of the Multifractal Detrended Fluctuation Analysis ([93, 55, 82] and references in [80]).

Our first simple task is to detect if the distance is able to discriminate between the two groups. In order to fully answer to this question, for all RR and HRV signals from the complete datasets $\mathbf{gk} = \{gk_1, \ldots, gk_{40}\}$ and $\mathbf{nk} = \{nk_1, \ldots, nk_{90}\}$ we have computed the *distance matrix* whose entries are given by the pairwise distances between any two sequences. Classification of any given patient is then performed in two steps. First, by removing them from the complete set (such obtaining two classes denoted by gk_group and nk_group). For instance, for patient gk_i we have $gk\_group(gk_i) = \{gk_j | j \neq i\}$ and $nk\_group(gk_i) = \{nk_j | j = 1, \ldots, 90\}$. Second, by taking the minimum over the averaged distance from both remaining groups, denoted by d_{gk} and d_{nk}, respectively. In formulas on the example above:

$$d_{nk}(gk_i) =< d(gk_i, nk_j) >_{j=1,\ldots,90}$$

and

$$d_{gk}(gk_i) =< d(gk_i, gk_j) > \begin{array}{l} j = 1, \ldots, 40 \\ j \neq i \end{array}$$

The patient is predicted either healthy or hospitalized accordingly to the resulting minimum distance.

As also emerges from the entropy analysis, ECG signals of healthy people are more complex than those of hospitalized ones, which show some order due to the limitated variability. Thus, intuition suggests that for a similarity measure it should be easier to clusterize two hospitalized signals to each other than two healthy series. Notwithstanding, the distance between two healthy patients may be smaller than that between two hospitalized ones.

Going into details concerning the results of the classification, in the RR signals, the lack of order already observed in the entropy values within the same class is even emphasized when observing the distances. In our experiments on all 40 **gk** and 90 **nk** sequences, there were 87 badly classified patients and they all were hospitalized subjects selected as healthy. Roughly speaking, the predictive method answers to the question whether a patient is hospitalized with a sensitivity (how many predicted not healthy are really hospitalized) of 3.3%. Definitely, one cannot be confident in the classification over plain RR series, even if the specificity (how many predicted healthy are really healthy) of the test is of 100%.

The landscape concerning distance classification on HRV signals is much more pleasant. Quantitatively, the improvement achieved w.r.t. RR codings is remarkable: there were only 21 incorrect classifications over the complete dataset. In this case, the sensitivity is $Sn = 77.8\%$ and the specificity is $Sp = 97.5\%$.

For what concern the HkV encoded data, when the distance $d(\cdot, \cdot)$ is used for data clustering, no appreciable improvements in the performance are observed for $k > 1$ (data not shown). For instance, in the case of H3V the test about hospitalized patients'prediction on the complete **gk** and **nk** datasets gives a sensitivity $Sn = 68.9\%$ and a specificity $Sp = 100\%$, thus confirming that just taking the *first discrete derivative* $k = 1$ seems to be the optimal procedure

After realizing that our method works quite well and consistently on the

full 24 hours ECG files, we then tested the distance on much shorter portion of the signals. In particular, using again the data from [80], we repeated the same experiments on the two 5 hours portion of the signals that correspond in general to the patients being awake (**nk_w**, and **gk_w** group) or at sleep (**nk_s**, and **gk_s** group), respectively.

The results obtained using the signals belonging to the wake state groups (**nk_w**, **gk_w**) are basically identical (if not better) to those obtained previously using the whole signal (see Table 3.3(b) where the subjects are the same of those in Table 3.3(a)). It is important at this point to remark that the binary sequences corresponding to these 5 hours interval are very small (few K bytes), which clearly implies a very limited set of words created during the parsing rule. In our opinion, the good performance of the method also on these very short sequences represents a clever indication of its consistency and efficacy.

To visualize the *clustering* property of our distance, we show in Fig. 3.10 the tree generated by the distance matrix computed using the group **gk_w** and 39 patience randomly chosen from the other group. It is important to remark that we use the tree to present our data only for exposition purposes and qualitative preliminary considerations. Quantitative and statistical features of our method have been in fact directly extracted from the numerical values of the distance matrix.

According to our expectations, the same experiments on the *sleeping* part of the data (**nk_s**, **gk_s** groups) give sometimes worst results, confirming that the wake part of the signal is evidently the most significant one.

3.3.2 The nsr-chf dataset

We proved the task of detecting healthy from unhealthy patients, with a different dataset of gk-nk.

We used the **nsr-chf** dataset described in section 3.2.1

As a common feature to all the experiments, also in this case we can note significant differences in the distances starting from the second digit.

Furthermore, we can also remark that the gap in the second digits become smaller when relative distances between healthy subjects (**nsr**) are computed, rather than relative distance between patients of the second group (**chf**). To say differently, healthy patients signals are more *similar* to one another than patients with past cardiac events.

In this case the sensitivity with the mean method attribution, is $S_n = 74\%$ and the specificity $S_p = 85\%$.

Moreover, also for these experiments is remarkable to note that the whole 24 hours signals of both groups could be substituted by the 5 hours corresponding to the wake period without degrading the final results (the **nsr_w** and **chf_w** dataset).

The final outcome of this experiment on wake signals are shown in Table 3.5 and in Fig. 3.11 where we can see that all the attributions are right.

3.3.3 The young-old dataset

A second kind of experiment that we have performed consists in clustering old patients from young patients, again by measuring the relative distances between the binary HRV coding extracted from the ECG signals. We have considered the **old_gk-young_nsr** dataset group (see section 3.2.1)

Also in this case, we consistently got correct results: a single young (old) patient has an averaged distance from the whole group of young (old) consistently smaller than the other group.

In order to give a visual presentation of part of the results, we again show a table with the averaged distances from the two groups (Table 3.5(a)).

We have then repeated the same experiment with the data downloaded from [53], where we have chosen some healthy individuals of age from 60 to 68, and other from 28 to 40 years (the **nsr_old-nsr_young** dataset). On this dataset we obtain only an error, the old patient nsr042 is attributed to the young class, but the significant differences in the distances starting from the fourth digit; infact $d(nsr042, old\_group) = 0,9496$ while $d(nsr042, young\_group) = 0,9494$. It is possible to observe this fact also from Fig. 3.12. Again, even if

the statistical significance of the results can be disputed due to the non high numbers involved, the portion of successes is very high and strongly support the validity of the method.

Finally, we use the last dataset of young-old problem: the **fantasia** dataset (see always section 3.2.1).

We remark that these signals are very short (about 14 K), but also in this case the discrimination with the method is able to distinguish in a satisfactory way the olds from youngs, obtaining an accuracy=75%.

3.3.4 The NYHA classification

In the last experiment we tried to recognize the NYHA class of individual patients with classified congestive heart failure. The NYHA (New York Heart Association) classification is a functional and therapeutic classification for prescription of physical activity for cardiac patients. The four classes are characterized in the follow way:

Class I: patients with no limitation of activities; they suffer no symptoms from ordinary activities.

Class II: patients with slight, mild limitation of activity; they are comfortable with rest or with mild exertion.

Class III: patients with marked limitation of activity; they are comfortable only at rest.

Class IV: patients who should be at complete rest, confined to bed or chair; any physical activity brings on discomfort and symptoms occur at rest.

After the usual binary coding, we repetitively chose few signals out of each class and use them as reference data for classification. We then picked randomly other unknown HRV strings and used the minimum averaged distance from the previously defined sets to attribute the NYHA classification. Our method is able to distinguish quite well the subjects in class I and III,

whereas quite often it consistently attributes to class III patients that were classified in class II. Because of the small number of trials, the numerical results are not statistical significant and are not presented here. In any case this classification is quite subjective, being related to the general conditions of the patience, and it is not surprising if it will turns out to be difficult to detect a sharp boundary between class II and III of the NYHA classification using our or others statistical methods.

3.4 Comparison with standard analysis

We now briefly compare our methods with standard discrimination analysis techniques. It is worth mentioning, as already addressed by others in [111], that even if sometimes the similarity distance used here seems to reach a slightly better performance than other standard techniques in discriminating the two groups patients, *this is not* the main point we want to address here. We believe that in any case this or other similarity measures offer new methods for quantitative estimates of the relative information content between groups of symbolic strings (not only coded ECG), allowing data clustering or phylogenetic tree construction in a novel way with respect more traditional approaches.

The aim is to compare our results with the ones obtained by implementing standard discriminant techniques analysis, along the lines of [45].

More precisely, for each RR signal we calculate the mean and variance over the interbeat times and we try to see whenever these pair of indicators are robust for classification into the two groups.

We show here the results on the **gk-nk** dataset. First of all, the mean by itself is not sufficient to discriminate efficently beetween the two groups, as qualitatively indicated in Fig.3.4, where the distributions of the mean along the two groups are shown.

Furthermore, in Fig.3.3 we plot all the data in the plane, where we have put the mean and the variance on the x and y axes respectively.

In figure 3.5 we have plotted the mean and variance for the gk and a subset of nk, the more difficult to clustering (as we can seen from the figure 3.3 there is a subset of the 90 nk for which the values of mean and std are visibly different from those of the gk dataset)

In figures 3.6 the result of the PCA analysis on the same patients of figure 3.5. We can see that the PCA does not change considerably the clusteritazion task.

From the quantitative point of view, standard discrimination analysis performed on this set of data reach a quite positive performance but overall, at least for our dataset, the similarity distance allows to reach a slightly better performance.

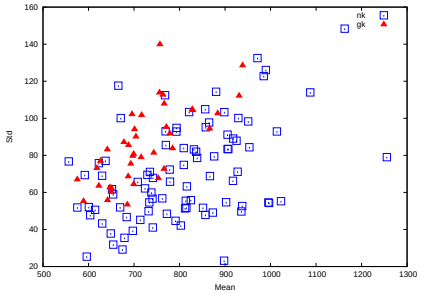


Figure 3.3 ◇ Plot of the mean (x-axis) and variance (y-axis) for the RR sequences in the **gk** (triangle) and **nk** (square) group.

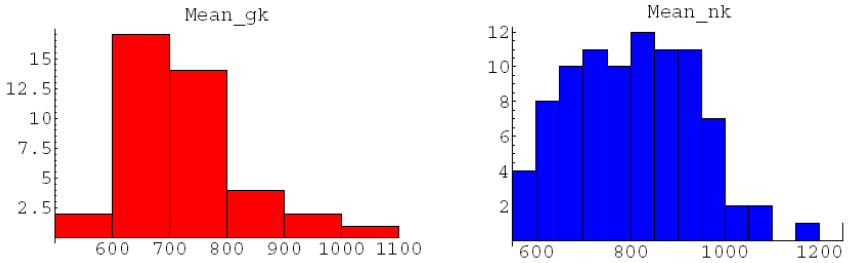


Figure 3.4 ◇ Distribution of the mean values of interbeat times (in milliseconds on the x axis) for the RR sequences in the **gk** (left) and **nk** (right) group.

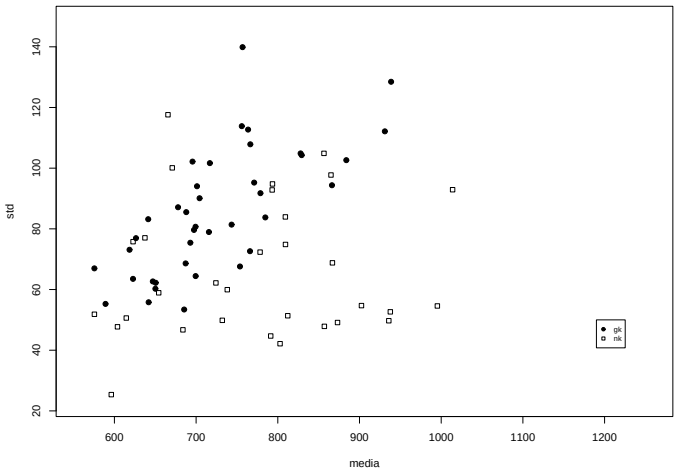


Figure 3.5 ⋄ Plot of the mean (x-axis) and variance (y-axis) for the RR sequences in the **gk** (star) and some patients of the **nk** (square) group.

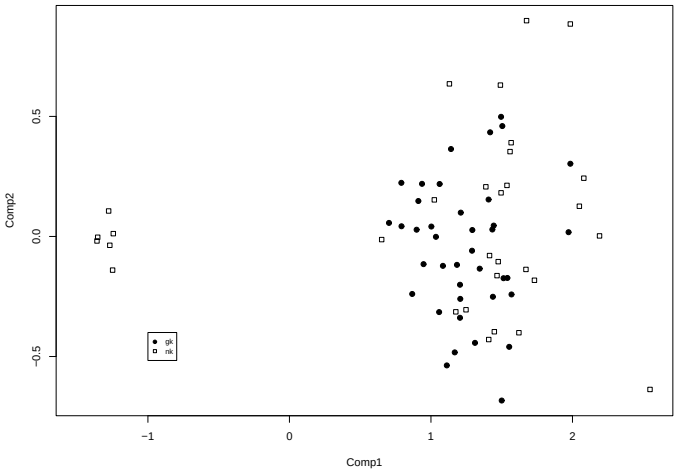


Figure 3.6 ⋄ PCA analysis on the gk group and share of the nk group.

3.5 n-gram distance on heart signals

In this last section we apply the n-gram distance (defined in section 2.2.6) on the sequences belonging to the **gk-nk** dataset, and also to some of the previously datasets.

From each HRV sequence S, we can fixe a value of n and calculate all distinct words of length n that can be founded in the sequence. Our original sequence S now can be written as a vector
$S = \{\{w_1, f_S(w_1)\}, \{w_2, f_S(w_2)\}, \ldots, \{w_N, f_S(w_N)\}\}$, where w_i is the i-th word of length n in S, and $f_S(w_i)$ is the frequency of this word in the whole sequence.

As have already argumented Yang at al. in [111], the idea is that the occurence of these $n-$words reflects the underlying dynamics of the original time series. Different types of dynamics thus produce different distributions of these $n-$words.

3.5.1 Attribution Methods

Also in this case we have used the attribution methods defined in the previous section (see section 3.2.2). We have tested the performance of the n-gram distance also with another standard indicator of the sensitivity and specificity of an algorithm: the ROC curve [83].

In a Receiver Operating Characteristic (ROC) curve the true positive rate (Sensitivity= $\frac{TP}{TP+FN}$) is plotted in function of the false positive rate (=$\frac{FP}{FP+TN}$=1-specificity) for different cut-off points. Alternatively, one can also display the sensitivity versus the specificity (=$\frac{TP}{TP+FP}$). Each point on the ROC plot represents a sensitivity/specificity pair corresponding to a particular decision threshold. A test with perfect discrimination (no overlap in the two distributions) has a ROC plot that passes through the upper left corner (100% sensitivity, 100% specificity). Therefore the closer the ROC plot is to the upper left corner, the higher the overall accuracy of the test.

The accuracy of the test depends on how well the test separates the group

being tested into those with and without the disease in question. Accuracy is measured by the area under the ROC curve (AUC). An area of 1 represents a perfect test; an area of 0.5 represents a worthless test.

For applying the ROC analysis to our distance matrices we must convert a matrix of dimension $n \times n$ in a vector of length $n \times n$ and used this vector as prediction. The class label is obtained putting in a vector the similarity matrix with entries 1 if the two protein domains belong to the same group, 0 otherwise.

Moreover, we have used a cluster algorithm on our matrices in order to asses the performance of the n-gram based classification. The cluster algorithm used is PAM a version of the K-means algorithm, as implemented in R-package [104].

We have performed the ROC Analysis using the ROCR package [100], while, for the PAM algorithm we have used the CLUSTER package [104].

The use of the ROC analysis follows the more standard way for assessing the intrinsic ability of the metodology to discriminate and classify cardiological sequences. The cluster algorithm on the matrix is, insteed, a classical procedure applied to similarity matrices.

3.5.2 Results

In Table 3.3 and Table 3.2 we resume the outcomes obtain with the geometric formula (54), the LZ-distance and the rank formula, according the three different attribution method explained in section 3.2.2. The dataset used is the **gk-nk dataset**.

The results that we will show are obtained using the geometric formula (54) with $n = 10$, but with n from 6 to 12 we can obtain similar results. This values of n agrees also with the results of [111] where they use words of length equal to 8.

Following the authors in [111, **?**], we remember the rank distance:

$$D_n(S_1, S_2) = \frac{1}{2^n - 1} \sum_{k=1}^{2^n} \mid R_1(w_k) - R_2(w_k) \mid F(w_k) \tag{1}$$

where $F(w_k) = \frac{1}{Z}[-p_1(w_k)\log p_1(w_k) - p_2(w_k)\log p_2(w_k)]$, and $p_1(w_k)$, $R_1(w_k)$ represents probability and rank of a specific word.

Respect to the formula in [111], our distances consider shared and no shared words: we believe, that a word present in a signal and not in another is a word indicanting a peculiarity for the signal indeed. The high percentage of success that we can have also for values of $n > 10$ (while the formula in [111] have a decrease in performance) is an indicator of the existence of long pattern in the sequences that can characterize the sequence of an healthy subject rather than an unhealthy subject. So, the not shared words become "discriminant" words between a group and another.

In figure 3.7, infact, we can see that similar results are obtained for $n < 10$. After this value, the rank formula (1) have an abrupt decrease in success, while our formulas continue to have good results untill $n = 20$.

About the ROC and CLUSTER analysis, in figure 3.8(a) we have plotted the AUC values for different values of n and different formulas:

- the formula (53) with $\alpha = 1$
- the formula (53) with $\alpha = 2$
- the formula (53) without the normalization factor, equivalent to the Euclidean Distance
- the geometric formula (54)

In these analyses we have included also other two dataset, beyond the **gk-nk** dataset: the **nsr-chf** dataset and the **nsr_w chf_w** dataset, other datasets of healthy/unhealthy patients (see figures 3.8(b) and 3.8(c)).

Otherwise, in figure 3.9 we plotted the percentages of achievement obtained with the above formulas on the clusterization task, for the same tree different datasets.

We can observe that the behavior of the formula (53) with $\alpha = 1$ is similar to that of the geometric formula. The Eulidean distance has a poor performance respect to the other distances, while the formula (53) with $\alpha = 2$ has different behavior according different datasets.

For the **gk-nk** dataset we can see an improvment in the percentage for values of n bigger than 15; if we analyze the dictionary of the signals, we can see that, in average, this value of n is the same for which appear words in the dictionary of healthy patients that does not appear in the dictionary of unhealthy signals.

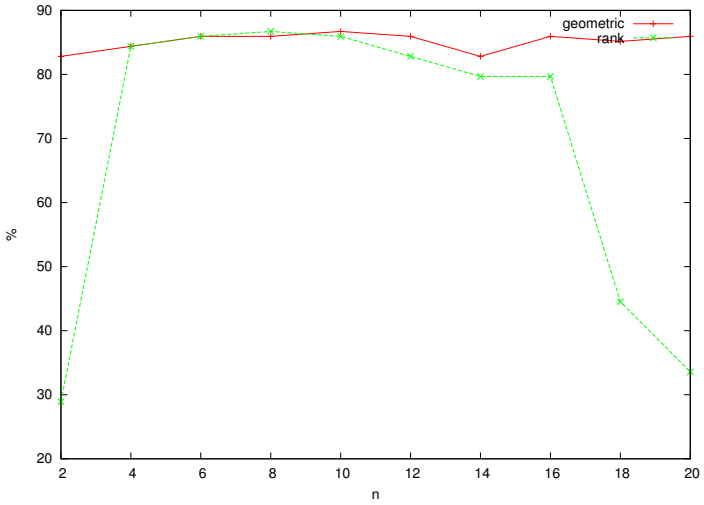


Figure 3.7 ◇ Comparison of the geometric formula and the rank formula; in the x axis the value of n in the y axis the percentage obtained by the distances with the nearest neighboor method

RR200	n	$c(x)$	h
gk28_w_200	27770	6252	0,412964
gk30_w_200	25377	5475	0,416421
nk13_w_200	24558	5360	0,434764
gk19_w_200	27634	6738	0,439458
nk11_w_200	26050	6246	0,459744
gk03_w_200	25863	6202	0,459868
gk11_w_200	25795	6283	0,460132
nk35_w_200	23392	5414	0,466981
gk17_w_200	23761	5974	0,47064
nk19_w_200	23516	6153	0,478126
gk36_w_200	22769	6096	0,493595
gk09_w_200	23340	6443	0,497071
nk32_w_200	21401	5747	0,505378
gk13_w_200	24390	7618	0,506991
nk15_w_200	22033	6293	0,510834
gk37_w_200	20669	5630	0,520561
nk09_w_200	19712	5213	0,524032
nk10_w_200	17169	4289	0,531983
nk22_w_200	19223	5493	0,539087
nk38_w_200	18706	5245	0,595519

HRV	n	$c(x)$	h
nk09_w	19711	1128	0,81644
nk22_w	19222	1103	0,816575
nk11_w	26049	1469	0,827236
nk13_w	24557	1409	0,836773
nk38_w	18705	1108	0,840619
nk35_w	23391	1366	0,847577
nk32_w	21400	1275	0,857069
nk19_w	23515	1399	0,863929
nk15_w	22032	1320	0,864381
gk09_w	23339	1391	0,86482
gk37_w	20668	1248	0,8656
nk10_w	17168	1059	0,867743
gk28_w	27769	1634	0,868587
gk30_w	25376	1512	0,871782
gk11_w	25794	1540	0,874944
gk17_w	23760	1441	0,881596
gk03_w	25862	1570	0,889874
gk19_w	27633	1679	0,896469
gk36_w	22768	1412	0,897677
gk13_w	24389	1520	0,908294

Table 3.1 $\diamond$ Entropy values h, in increasing order, for a subset of 20 patients (see text), together with corresponding complexity $c(x)$ and length n of RR[200] (top table) and HRV codings (bottom table), respectively.

(a) Mean

Rank	87,18%
LZ	88,46%
10Geo	89,74%

(b) Nearest Neighbor

Rank	85,90%
LZ	79,49%
10Geo	83,33%

(c) Vertical Voting

Rank	85,90%
LZ	91,03%
10Geo	87,18%

Table 3.2 ⋄ Percentages of achievement obtained with the above distances: Rank is the distance defined in [111] with n=8, LZ is the distance seen in section 2.1.3, 10Geo is the distance in equation (54) with n=10. The dataset (cardiological signals) has 40 healthy patients and 40 unhealthy ones.

(a) Mean

Rank	85,16%
LZ	84,38%
10Geo	85,94%

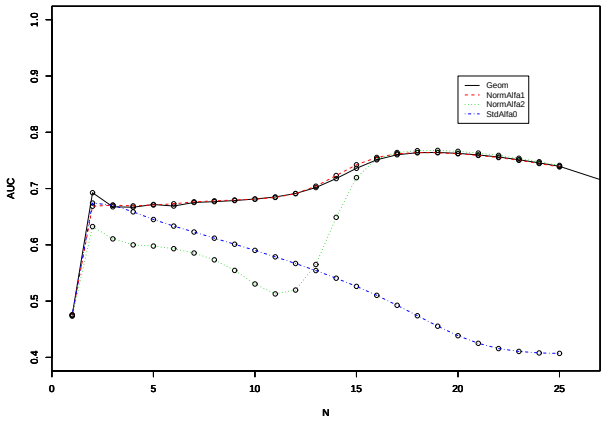
(b) Nearest Neighbor

Rank	81,25%%
LZ	85,16%
10Geo	86,72%

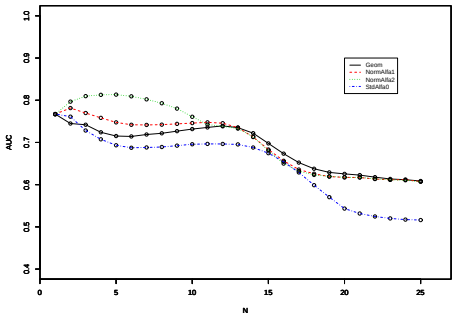
(c) Vertical Voting

Rank	85,16%
LZ	85,94%
10Geo	86,72%

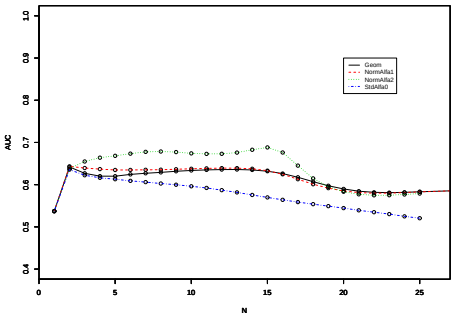
Table 3.3 ⋄ Comparison of the percentages of achievement obtained with the same distances of the above tables. The dataset is composed by 40 gk and 90 nk (cardiological signals)



(a) In the x-axis the value of n and in the y-axis the AUC value obtained by different n-grams formulas on the **gk-nk** dataset

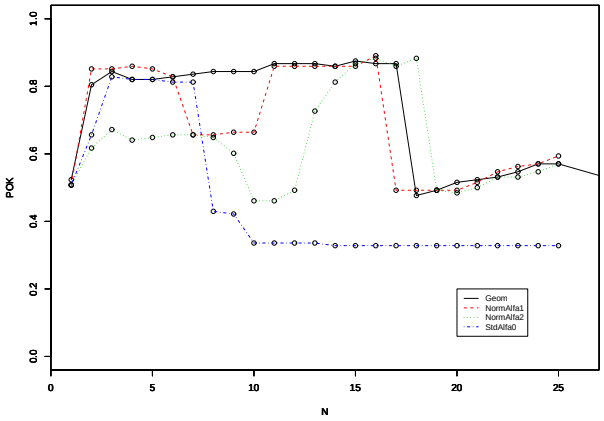


(b) In the x-axis the value of n and in the y-axis the AUC value obtained by different n-grams formulas on the **nsr-chf** dataset

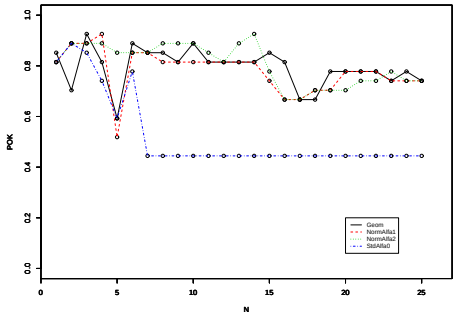


(c) In the x-axis the value of n and in the y-axis the AUC value obtained by different n-grams formulas on the **nsr_w-chf_w** dataset

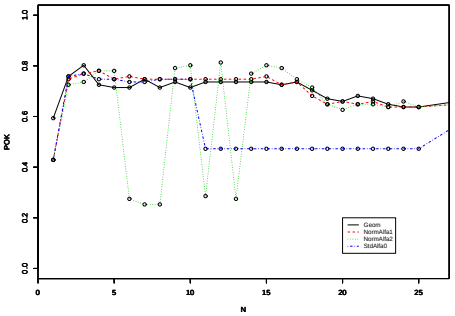
Figure 3.8 ⋄ Plots of the AUC values for different datasets. In all graphics the formulas in legend are: Geom= Geometric formula (54), NormAlfa1 = the formula (53) with $\alpha = 1$, the NormAlfa2= the formula (53) with $\alpha = 2$ and StdAlfa0= the formula (53) without the normalization factor and with $\alpha = 0$ (the euclidean formula)



(a) In the x-axis the value of n and in the y-axis the percentage of success obtained by the clustering procedure with different n-grams formulas on the **gk-nk** dataset



(b) In the x-axis the value of n and in the y-axis the percentage of success obtained by the clustering procedure with different n-grams formulas on the **nsr-chf** dataset



(c) In the x-axis the value of n and in the y-axis the AUC value obtained by different n-grams formulas on the **nsr_w-chf_w** dataset

Figure 3.9 ⋄ Plots of the percentage of success obtained with the PAM clustering algorithm for different datasets. In all graphics the formulas in legend are: Geom= Geometric formula (54), NormAlfa1 = the formula (53) with $\alpha = 1$, the NormAlfa2= the formula (53) with $\alpha = 2$ and StdAlfa0= the formula (53) without the normalization factor and with $\alpha = 0$ (the euclidean formula)

(a) full 24 hours ECG signals

	gk_group	nk_group
gk02_nn	0,950977	0,955649
gk03_nn	0,9512	0,959749
gk04_nn	0,951591	0,957155
gk05_nn	0,949889	0,953167
gk06_nn	0,949679	0,958141
gk07_nn	0,951273	0,962977
gk08_nn	0,951308	0,962828
gk09_nn	0,949684	0,95644
gk10_nn	0,950085	0,959365
gk11_nn	0,949688	0,954517
gk13_nn	0,94936	0,95906
gk14_nn	0,949817	0,957204
gk15_nn	0,951751	0,964054
gk16_nn	0,949499	0,952967
gk17_nn	0,950058	0,956208
gk18_nn	0,951352	0,958267
gk19_nn	0,950012	0,957825
gk20_nn	0,953429	0,965333
gk21_nn	0,950678	0,959302
gk22_nn	0,950278	0,958852
nk10_nn	0,953073	0,952105
nk11_nn	0,955284	0,950414
nk12_nn	0,951612	0,954686
nk13_nn	0,955527	0,950697
nk14_nn	0,95358	0,958575
nk15_nn	0,952657	0,950346
nk16_nn	0,95545	0,952969
nk17_nn	0,975155	0,969354
nk18_nn	0,976497	0,964703
nk19_nn	0,952482	0,950202
nk20_nn	0,960154	0,955664
nk21_nn	0,960711	0,95591
nk22_nn	0,956478	0,95132
nk23_nn	0,961284	0,959017
nk24_nn	0,949156	0,956412
nk25_nn	0,959659	0,957893
nk26_nn	0,966242	0,958213
nk27_nn	0,960459	0,952844
nk28_nn	0,950147	0,953585
nk29_nn	0,953256	0,953296
nk30_nn	0,95347	0,953471

(b) wake part of ECG signal

	gk_group	nk_group
gk02_ w	0,944999	0,949697
gk03_w	0,942169	0,949849
gk04_w	0,94477	0,949449
gk05_w	0,946066	0,947472
gk06_w	0,943874	0,953748
gk07_w	0,945075	0,960126
gk08_w	0,94387	0,955866
gk09_w	0,943006	0,951416
gk10_w	0,941327	0,954052
gk11_w	0,942418	0,945749
gk13_w	0,940751	0,948664
gk14_w	0,942632	0,954633
gk15_w	0,943504	0,956356
gk16_w	0,94459	0,947752
gk17_w	0,940355	0,949688
gk18_w	0,944521	0,950204
gk19_w	0,942666	0,946773
gk20_w	0,944984	0,960437
gk21_w	0,943947	0,955633
gk22_w	0,944009	0,95303
nk10_w	0,94555	0,94192
nk11_w	0,950804	0,942961
nk12_w	0,94292	0,943463
nk13_w	0,950983	0,941804
nk14_w	0,949428	0,952428
nk15_w	0,947493	0,944664
nk16_w	0,950896	0,944168
nk17_w	0,970349	0,962885
nk18_w	0,964134	0,948842
nk19_w	0,946231	0,942469
nk20_w	0,948818	0,946029
nk21_w	0,966554	0,953678
nk22_w	0,953546	0,942596
nk23_w	0,960295	0,954839
nk24_w	0,943758	0,952151
nk25_w	0,953903	0,949325
nk26_w	0,961058	0,949702
nk27_w	0,966069	0,951091
nk28_w	0,944156	0,947916
nk29_w	0,949629	0,948452
nk30_w	0,945028	0,942885

Table 3.4 ⋄ Averaged distances of each patient from **gk_group** =(**gk**30, **gk**31,$\cdots$, **gk**39) and **nk_group**=(**nk**30, **nk**31,$\cdots$, **nk**39), respectively. Wrong classifications are marked in *red*.

	chf	**nsr**
chf01_w	0,988736	0,993407
chf02_w	0,992512	0,994858
chf03_w	0,971186	0,996126
chf04_w	0,980403	0,991931
chf05_w	0,980736	0,992299
chf06_w	0,979843	0,9914
chf07_w	0,974151	0,993553
chf08_w	0,994647	0,99748
chf09_w	0,969402	0,994815
chf10_w	0,966486	0,992431
chf11_w	0,979891	0,99794
chf12_w	0,981962	0,992295
chf13_w	0,973136	0,996432
nsr01_w	0,994181	0,925976
nsr02_w	0,993675	0,928663
nsr03_w	0,993803	0,923911
nsr04_w	0,994018	0,935523
nsr05_w	0,994254	0,925418
nsr06_w	0,994561	0,930583
nsr07_w	0,993325	0,922587
nsr08_w	0,994585	0,938982
nsr09_w	0,994489	0,923555
nsr10_w	0,994857	0,926272
nsr11_w	0,994628	0,92443
nsr12_w	0,994004	0,931252
nsr13_w	0,994587	0,923272

Table 3.5 $\diamond$ Averaged distances obtained by comparing non healthy patients (**chf**), with healthy subjects (**nsr**). For any single individual, the average is calculated among all the other patients in each group. Wrong classifications are marked in *red*.

(a) Averaged distances obtained by comparing young patients [53] (recorded with numbers) against old individuals (gk).

	Young	**Old**
16272	0,948901182	0,94897375
16273	0,949356636	0,952352833
16420	0,951251364	0,95512975
16483	0,957391273	0,95428375
16539	0,955175818	0,9531345
16773	0,954099182	0,952232333
16786	0,951502545	0,953977667
16795	0,949578455	0,9527505
17052	0,950846545	0,955799333
17453	0,949297455	0,950581
18177	0,951735364	0,953157417
18184	0,954342545	0,951167333
gk11_nn	0,952366167	0,949516818
gk12_nn	0,951272333	0,949272273
gk13_nn	0,950989583	0,9476455
gk14_nn	0,950517167	0,9487745
gk15_nn	0,955528917	0,9496472
gk16_nn	0,948811833	0,9510059
gk17_nn	0,95274075	0,9486855
gk18_nn	0,955450167	0,9518555
gk19_nn	0,953471583	0,9502753
gk20_nn	0,95707775	0,9512193
gk21_nn	0,952414667	0,9481727
gk22_nn	0,95289925	0,9486232

(b) Averaged distances obtained by comparing young patients (recorded with **f$y**) against old individuals (**f$o**). The signals came from the **fantasia** dataset

	old	young
f1o01	0,944046	0,944339
f1o02	0,939766	0,949769
f1o03	0,940316	0,94514
f1o04	0,939967	0,947402
f1o05	0,944787	0,953999
f1o06	0,951369	0,954935
f1o07	0,941646	0,940002
f1o08	0,936127	0,943446
f1o09	0,936086	0,94228
f1o10	0,937588	0,945268
f1y01	0,958599	0,948916
f1y02	0,943743	0,939759
f1y03	0,950451	0,948954
f1y04	0,953413	0,961138
f1y05	0,944913	0,947495
f1y06	0,940887	0,938414
f1y07	0,941744	0,93958
f1y08	0,938182	0,938462
f1y09	0,941773	0,939439
f1y10	0,943689	0,938555
f2o01	0,952783	0,947364
f2o02	0,952353	0,954606
f2o03	0,943765	0,953155
f2o04	0,964909	0,958742
f2o05	0,94226	0,951906
f2o06	0,93807	0,938358
f2o07	0,939695	0,944959
f2o08	0,944708	0,948949
f2o09	0,942533	0,951044
f2o10	0,944294	0,946527
f2y01	0,944858	0,944622
f2y02	0,95231	0,944257
f2y03	0,958638	0,962726
f2y04	0,954157	0,952182
f2y05	0,946311	0,945123
f2y06	0,943546	0,947424
f2y07	0,958203	0,950327
f2y08	0,9534	0,955053
f2y09	0,943919	0,944201
f2y10	0,948729	0,943685

Table 3.6 ◇ Results on young-old task. For any single individual, the average is calculated among all the other patients in each group. Wrong classifications are marked in *red*.

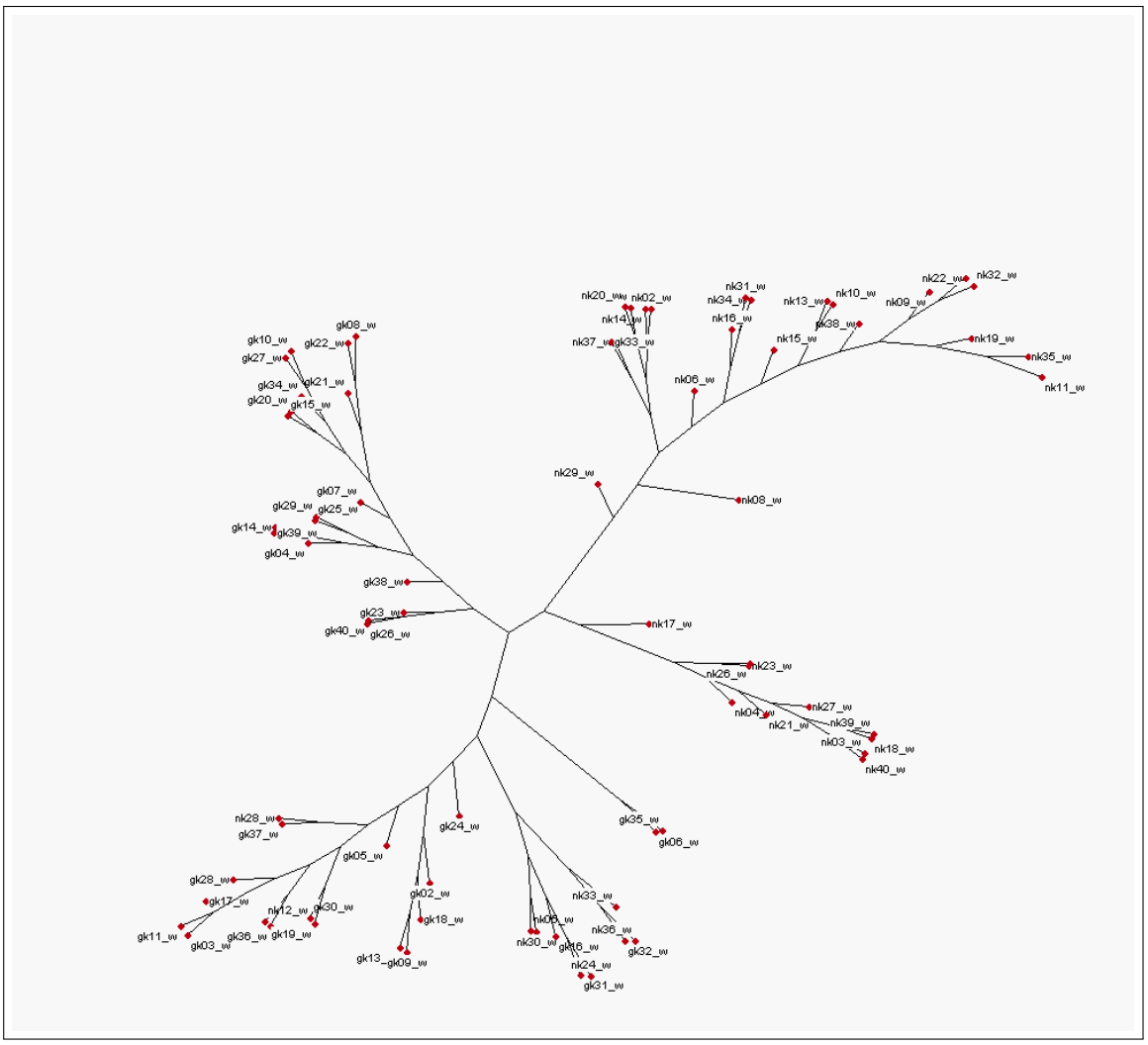


Figure 3.10 ⋄ Tree generated by the distance matrix computed using the group **gk_w** and 38 patience randomly chosen from the **nk_w** group.

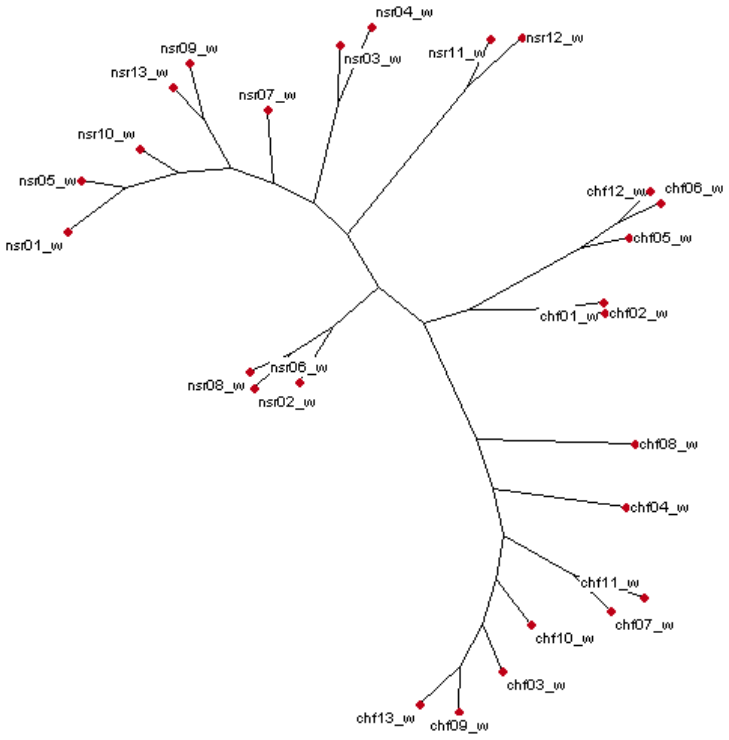


Figure 3.11 ⋄ Distance tree out of subjects from the **chf** and **nsr** groups.

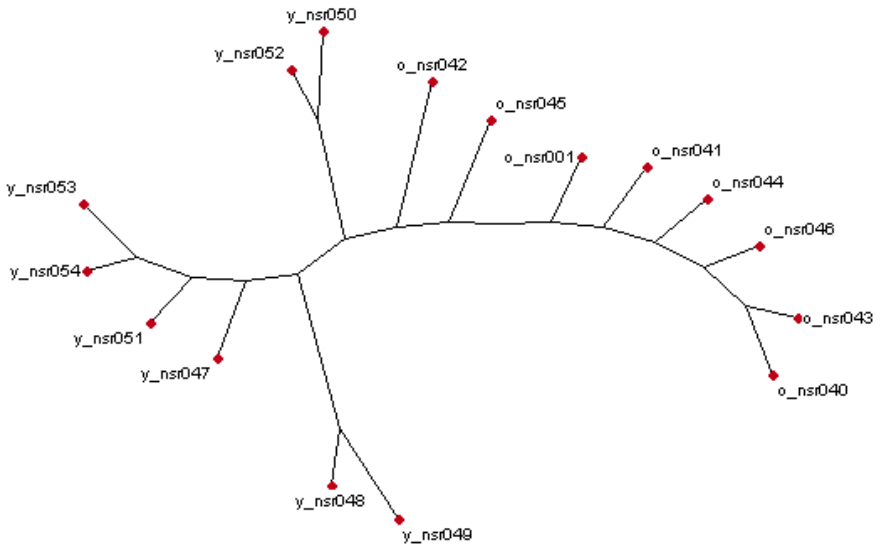


Figure 3.12 ⋄ Distance tree based on two groups of old and young subjects. The patients are the ones of the **nsr_old-nsr_young** dataset

Chapter 4

Protein sequences

In this chapter we introduce some of the methods used on protein sequences in order to classify proteins according to their structural class and to give, finally, some answers to the folding problem.
In the first part of the chapter some biological notions are recalled, principally those related to the protein similarity concept.
After describing some of the most known methodologies, in the final part of the chapter the introduced methods wil be compared by analyzing both their formulas and the literature's results.
This is done for the sake of finding the state of the art of methodologies that try to point out features marking a certain fold rather than others, starting from protein sequence analysis.

4.1 Biological concepts

Proteins are the main building blocks and functional molecules of the cell, taking up almost 20% of a eukaryotic cell's weight, the largest contribution after water (70%).

The building blocks for proteins are the *amino acid* molecules: there are 20 different amino acids that differ in the R side chains which determine their properties. As a convention each amino-acid is denoted by a letter in Latin alphabet. In figure 4.1 is shown the characteristic structure of an amino acid. The central carbon atom is called C_α to which are attached an atom of hydrogen, amino group, a carboxylic group and the R-group or *side chain*

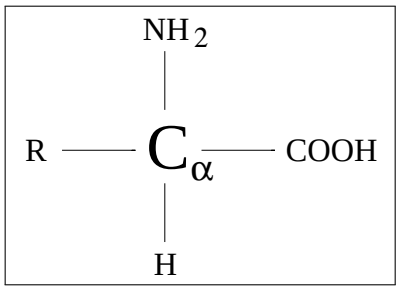


Figure 4.1 ⋄ amino acid

(see figure 4.1). The Proline is an exception to this structure because its side group links to the amino group.

The most peculiar features of the R-group are polarity and steric site, basically related to size and mobility of the residue. In table 4.1 are shown the 20 amino acid with their codes.

Two or more amino acids can be joined with a peptide bound that is a chemical bound formed between two molecules when the carboxylic group of one molecule reacts with the amino group of the other molecule, releasing a molecule of water (H_2O). Polypeptides and proteins are chains of amino acids held together by peptide bonds, whereas the atoms joined by the peptide bonds without the amino acid side chains are named *backbone* of the protein.

The sequence of the amino acid is known as the *primary structure* and it can be represented as a string of 20 different symbols. The length of the protein molecules can vary from few up to many thousands of amino acids.

Although the primary structure of a protein is linear, the molecule is not straight, and the sequence of the amino acids affects the folding. Precisely, the backbone of a protein is all laid on a plane: only 2 bonds can freely rotate, the $C_\alpha - N$ bond along the angle ϕ and $C_\alpha - C(O)$ bond along the ψ angle (see figure 4.2). The possible ϕ and ψ values are constrained by the structure of the adjacent amino acid residues.

According to the values of the angles ϕ and ψ, the protein can assume a

Amino Acid	**3-letters code**	**1-letter code**
Alanine	Ala	A
Arginine	Arg	R
Asparagine	Asn	N
Aspartic acid	Asp	D
Cysteine	Cys	C
Glutamic acid	Glu	E
Glutamine	Gln	Q
Glycine	Gly	G
Histidine	His	H
Isoleucine	Ile	I
Leucine	Leu	L
Lysine	Lys	K
Methionine	Met	M
Phenylalanine	Phe	F
Proline	Pro	P
Serine	Ser	S
Threonine	Thr	T
Tryptophan	Trp	W
Tyrosine	Tyr	Y
Valine	Val	V

Table 4.1 ⋄ List of standard amino acids

different configuration in the space, that is named the *secondary structure*. There are two common substructures often seen within folded chains: *alpha-helices* and *beta-strands*. They are typically joined by less regular structures, called *loops*.

Both these structures constitute around 50% of the protein structure. It is important to note how the α helix is really stabilized by local sequence interactions, typically between a radical and its 4^{th} neighbour. On the other hand β strand gives hydrogen bounds with β strand being spatially close

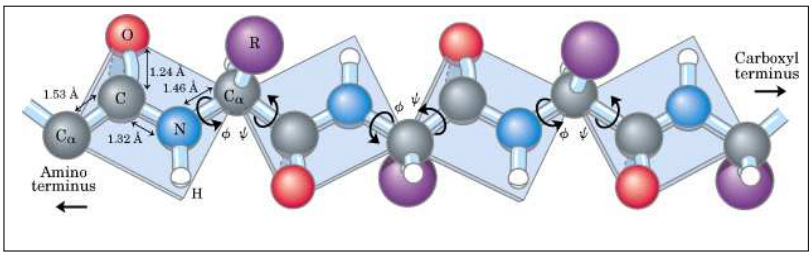


Figure 4.2 ⋄ amino acid chain

(but they may be quite far in the sequence).

There are striking regularities in the ways secondary structures are assembled [17]. These regularities arise from the intrinsic physical and chemical properties of proteins and provide the basis for the classification of the protein folds.

As a matter of fact, the different folds are usually grouped into classes (also named *structural classes*) on the basis of the secondary structures which they are composed of. In the SCOP dataset[1], for example, the folds are grouped in five classes:

- all alpha: for proteins whose structure is essentially formed by $\alpha-$helices
- all beta: for those whose structure is essentially formed by $\beta-$sheets
- alpha and beta: for proteins with $\alpha-$helices and $\beta-$strands argely interspersed
- alpha plus beta: for those in which $\alpha-$helices and $\beta-$strands are largely segregated
- multi-domain: for those with different folded domains and for which no homologues are known at present time.

As the results of the folding, parts of a protein molecule chain come into contact with each other and various attractive or repulsive forces (hydrogen

[1] we cite this dataset because it is used in most of the methods that we will see later. More information on this dataset will be given later on

bonds, disulfide bridges, attractions between positive and negative charges, hydrophobic and hydrophilic forces) between such parts cause the molecule to adopt a fixed relatively stable 3D structure, that is called *tertiary structure*.

The generation of the secondary structures previously described is a step in this folding process: every secondary structure can be seen as building block made up of a small number of amino acids that are close to one another, which then, in turn, interact, fold and coil to produce the tertiary structure that contains its functional regions (called domains).

At least, when a protein is formed from more than one chain of amino acids, it is said to have the *quaternary structure* (for example haemoglobin is made up of four chains).

4.2 Protein Similarity

In this section we want to clarify some concepts and biological terms that we will use later on.

First of all, the concepts of *similarity* and *homology*. Protein sequences, fold into unique three-dimensional (3D) structures. However, proteins with similar sequences adopt similar structures. Indeed, most protein pairs with more than 30 out of 100 identical residues were found to be structurally similar. This high robustness of structures with respect to residue exchanges explains partly the robustness of organisms with respect to gene-replication errors, and it allows for the variety in evolution.

As we have seen in the previous section, the function of a protein is strictly related to its structure. Two proteins are similar when they have similar structures. Rougly speaking, two structures of proteins are similar when we can overlap one on the other.

It is evident that it is necessary to decide a *measure* for quantify this similarity concept. In the section 4.2.2 we will show which kind of measures are used for this scope. Similarly to the procedure used in the assignment of similarity between sequences, once it is decided the procedure to align the

two structures it is even possible to define a score to quantify the difference in the alignments.

Proteins that share similar structures, may also have similar sequences. Usually, similarity between a pair of aligned biological sequences is measured as *sequence identity*: the number of aligned positions where the matching characters (e.g. amino acids in proteins) are identical. A more accurate analysis of the alignment sequence methods in the case of protein sequences is given in section 4.2.1. Nevertheless, as already underlined above, there exist proteins that have similarity in structure but a low percentage of sequence identity that indicates a poor sequence similarity.

From biological point of view, proteins that share similar structures (to which often similar functions are related) and a common ancestor are said *homologus* proteins. The existence of a common ancestor can be detected by the analysis of their sequences: an evolutionary relationship between a pair of sequences is usually inferred on the basis of high sequence identity between them. Moreover, common fold means that sequence identity can be used to identify proteins with similar three-dimensional structures. Then, while the common structures can be detected by using alignment structures algorithms, it is also necessary to use some alignment sequences algorithms for searching a common evolutionary origin.

Therefore, whenever two proteins sequences or protein structures seem very similar, the similarity can be explained by two different alternative: the two proteins are similar because they are homologous, i.e. both are descendants from a common ancestor, or the proteins are not related, i.e. they are similar because some set of structural or functional constraints caused them to converge from independent origins to the observed similarity. If the observed similarity is sufficiently great that it seems unlikely for it to occur several times independently then this can be considered as a discriminating factor between homologous and analogous sequences. The inference of homology, is based on both the degree of similarity that they share (measured by the alignment score or the percentage of identity sequence) and some sense

of how unlikely it is that this similarity could have arisen independently [92]. Therefore, proteins with similar structures can be classified in two groups:

1. **homologous** proteins: these are proteins with similar structure likely to be the result of evolutionary divergence. So, they have also similarity in sequences. In particular these proteins are divided in remote, medium and close homology, sub-division based on the percentage sequence identity.

2. **analogue** proteins: are proteins with similar three-dimensional structures (same SCOP classification, and Topology in CATH) but little evidence of common ancestor (different SCOP superfamily classification).

The structure alignments, that we will introduce in section 4.2.2, unambiguously distinguish between protein pairs of similar and non-similar structure when the pairwise sequence identity is high ($> 40\%$ for long alignments). At the same time, accuracy of secondary structure prediction based on multiple sequence alignments, drops significantly when low homology sequences are considered.

The term homologous protein *families*, 15-20 years ago meant clusters of homologous proteins. Nevertheless, the greater sensibility of the comparison methods for amino acid sequences and the major number of tertiary structures discovered, has shown the evolutionary relationship between a lot families previously known.

Usually, the belonging of a protein to a particular family implies a specific biological function, whereby the name of the family is given.

Actually the number of known families is around ten thousand, for example in the Pfam dataset [39], there is a list of 8183 families.

Large families of homologous proteins are commonly subdivided in *subfamilies*, according to the comparison of the similarity level of their amino acid sequences. The evolutionary related families can be collected in *superfamilies* (or *clans*). In the Pfam dataset, for example, in 206 clans are

contained 1396 families.

A problem related to the distinction between families can be the complex structure of the *domain* in many proteins. Different regions along a single polypeptide chain can act as independent units, to the extent that they can be excised from the chain, and still be shown to fold correctly, and often still exhibit biological activity. These independent regions are termed domains. Domains of proteins can be detected through the analysis of the tertiary structure; the existence of experimental data on 3 dimensional structures allows to single out the number of domains and the border lines between them in the protein structure. Different domains have different biological functions; the domains are also called functional domains. The protein domain is the basic classification unit for the SCOP (Structural Classification of Proteins) database [85], a comprehensive ordering of all proteins of known structures, according to their evolutionary and structural relationships. Small proteins, and most of those with medium size, have a single domain and are, therefore, treated as a whole. The domains in large proteins are usually classified individually, indeed. Most of the datasets used in the methods that we will see later are subset of the SCOP [85] or CATH [87] databases.

The very first step for both these databases is to separate the proteins into domains. It is difficult to produce an unequivocal definition of a domain and this is one area in which CATH and SCOP differ.

Structural Classification of Proteins, or *SCOP*, was among the earliest efforts to classify protein structures into folds. Protein domains with no obvious sequence homology to other domains are defined and classified manually [85]. This database has been considered the standard for protein structure classification in many ways.

Class Architecture Topology Homologous (*CATH*) superfamily makes use of a combination of manual and automated procedures in defining and classifying protein domains. CATH relies on the consensus of three automated classification methods to break protein chains into domains. This approach is effective in defining the domains of 53% of the chains. The domains of

chains for which consensus is not reached are defined manually. Homology is defined largely in terms of sequence, but distant sequences matching with high structural similarity may be defined as being homologous.

4.2.1 Sequence similarity algorithms: BLOSUM and PAM matrices

The measure of similarity between two protein sequences used in the aligment algorithms is a kind of a generalization of the Hamming distance, since the similarity is proportional to a certain score computed by summing up the partial scores associated to the one-to-one pairwise alignment of the two sequences. In the Hamming case the partial score is binary: 1 when we have the same amino acid in the same position of the two sequences, 0 otherwise.

As already observed above, the concept of similarity between proteins involve the research of a sort of common origin from which the two proteins have evolved under the phylogenetic pressure that, via deletions, insertions and substitutions, may create the differences between the two proteins. So, we have to find out sequences that are not only identical to the assigned probe, but are also "similar" in a suitable sense. Here lies the need to use a more flexible score than a binary evaluation (hamming distance), and that evaluates the substitution of the amino acid i with j.

Rougly speaking, the alignment of proteins starts as the alignment of DNA sequences, putting the sequences $seq1$ and $seq2$ one over the other $\begin{matrix} seq1 \\ seq2 \end{matrix}$. However, the score is not computed directly by the matches and mismatched between the two sequences but it is inferred by an *a priori* model (the substitution matrix) that tries to take into account of the possible common origin.

The basic idea for building this reference model is to measure the correlation between two sequences; given a pair of "correlated" sequences we can measure the substitution frequency of $i \rightarrow j$ (assuming symmetry) p_{ij}, i.e. the probability that the amino acid i can be change with the amino acid j,

and compare it with the null hypotesis (random correlation or indipendent events) $p_i p_j$.

The score associated to the pair i, j of amino acids belonging to two proteins X and Y respectively, is defined as:

$$s(X(k), Y(k)) = s(i, j) = \log \frac{p_{ij}}{p_i p_j} \tag{1}$$

Following this idea several substitution matrices have been derived. Their main difference is relative to the alignment types used for computing the frequencies.

The PAM (Point Accepted Mutation)[26] matrices are based on global alignments of closely related proteins. The PAM matrix M, relating each amino acid to each of the other 19, with an evolutionary distance of 1, would have entries $m(i, j)$ that indicates the probability of change from one amino acid i to another amino acid j in homologous protein sequences with at least 85% identity during short-term evolution. One PAM matrix corresponds to an average change in 1% of all amino acid positions.

An M^k matrix, which estimates the expected probability of changes at a distance of k evolutionary units, is then obtained by multiplying the M matrix by itself k times. Each M^k matrix is then associated to the scoring matrix PAM^k, whose entries are obtained on the basis of the log ratio:

$$s(i, j) = \log \frac{m^k(i, j)}{p(i)p(j)}$$

where $p(i)$ and $p(j)$ are the observed frequencies of the amino acid.

An alternative approach to estimate target frequencies, and the corresponding log-odds matrices, has been advanced by Henikoff and Henikoff [48] with the BLOSUM Matrix (BLOck SUbstitution Matrix). They examine multiple alignments of distantly related protein regions directly.

In these multiple alignments they look at "blocks" of conserved sequences (that are assumed to be of functional importance within related sequences). Then the target and background frequencies $p(i, j)$ and $p(i)p(j)$ are calculated using the database Blocks which contain sets of proteins with a con-

trolled maximum rate of percent identity θ that defines the BLOSUM matrix, so that BLOSUM-62 refers $\theta = 62\%$ and so forth.

The i, j entry of the BLOSUM matrix is given by:

$$s(i,j) = \log \frac{p(i,j)}{p(i)p(j)} \tag{2}$$

and the global score of two sequences X and Y of length n is given by summing up the scores relative to each position:

$$S(X,Y) = \sum_{k=1}^{n} s(x_k, y_k) = \sum_{i,j} n(i,j) \log \frac{p(i,j)}{p(i)p(j)} = n \sum_{i,j \in \mathcal{A}} f(i,j) \log \frac{p(i,j)}{p(i)p(j)} \tag{3}$$

where $n(i,j)$ is the number of occurrences of the pair i, j inside the aligned sequences, and $f_{ij} = \frac{n_{ij}}{n}$ is the relative frequency of the pair i, j.

We underline that, due to the different methods used to build the matrices, for very correlated sequences we must use PAM with low numbers and BLOSUM with large numbers. The opposite holds for distant sequences.

Once the substitution matrix is obtained, the score of the alignment of two sequences is given by this matrix.

For example: let $seq1 = VDSCY$ and $seq2 = VESLCY$.

Then the score between these two sequences is obtained bringing them into alignment:

$$\begin{array}{cccccccc} seq1 & V & D & S & - & C & Y \\ seq2 & V & E & S & L & C & Y \end{array}$$

and looking for the corrisponding score for the pair i, j in the blosum matrix.

From the BLOSUM matrix as in figure 4.3 we obtain:

$$\begin{array}{cccccccc} seq1 & V & D & S & - & C & Y \\ seq2 & V & E & S & L & C & Y \\ & 4 & 2 & 4 & -11 & 9 & 7 \end{array}$$

and the final score is 15.

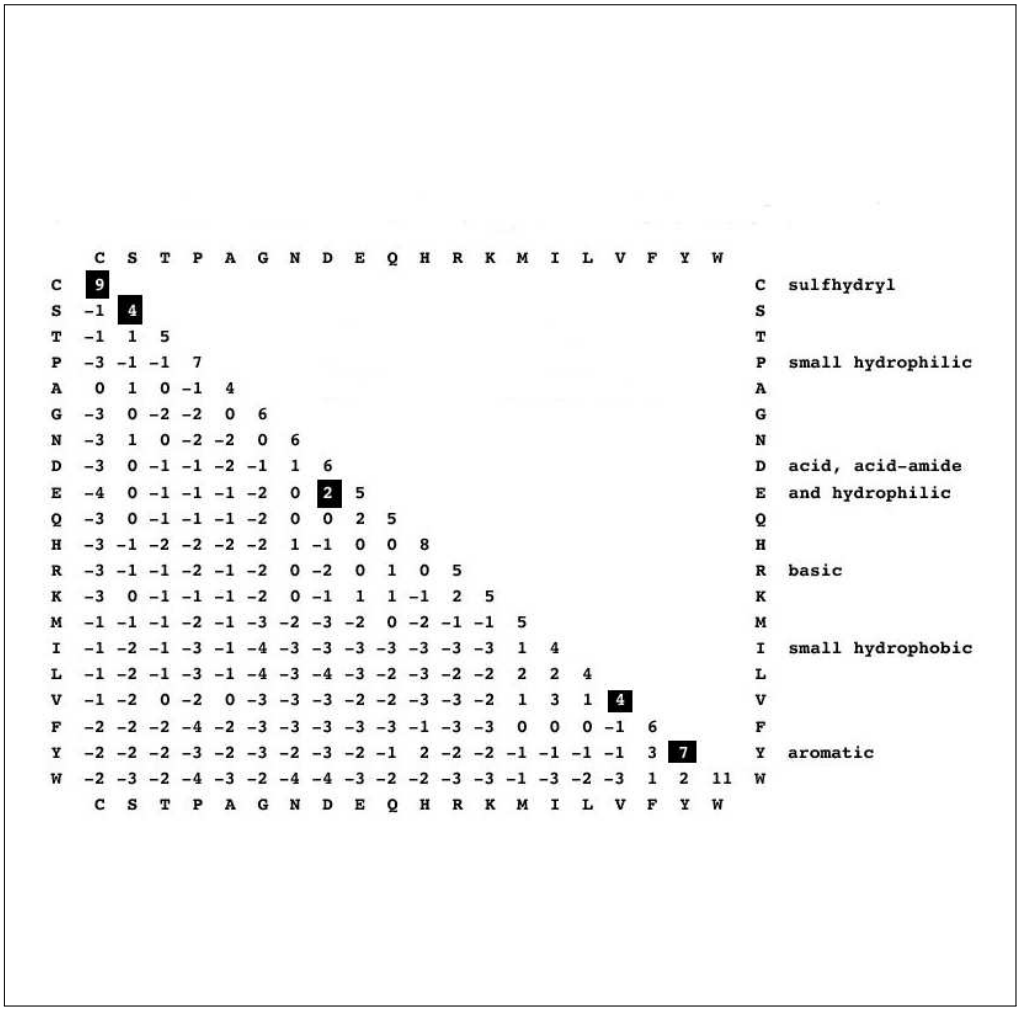


Figure 4.3 ⋄ Blosum62 amino acid substitution matrix. In the boxes the values corrisponding to the pair i, j in our example (see above).

4.2.2 Structure similarity algorithms

Differently from the sequences comparison, the similarity measures on protein structures take into account the coordinates of atoms. [2]

When comparing two protein structures, the algorithms search the similarity between the backbone geometries. As mentioned above, the backbone is a concatenation of atom triplets (N, C_α and C'), one for each residue in the protein. Due to the planarity of the peptide bond and the small variance of bond lengths and angles, the backbone geometry is completely determined by the positions of the C_α atoms. Therefore the protein structure can be represented as a sequence of 3-D points, specifying the centers of all C_α atoms.

The most direct approach to the comparison of two protein structures is to move the set of points representing one structure as a rigid body over the other, and look for equivalent residues. We can speak of a 'geometric' alignment. Here the problem of finding the optimal structural alignment between two protein structures is open, that is a NP-hard problem. The existing structural alignment algorithms use some simplification and reach out a reasonable, if not optimal, alignment. For a review of the most popular protein structure alignment we refer to [99].

There are also other approaches in wich the 3D geometry is ignored and the structures are compared using only the 3D profiles [102, 59]. For other references we refer to [69].

Two metrics are commonly used for the comparison [69]:

- **Coordinate root mean square deviation**: Given two sequence of points in 3D-space describing the C_α position of two protein structures $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and $Y = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$, the root mean square deviation (RMS) between the coordinates of the corresponding C_α (**cRMS**) is

[2] All the information on the coordinates of atoms are contained in the PDB files. The Protein Data Bank (**PDB**) [6] is a repository for 3-D structural data of proteins and nucleic acids. These data are typically obtained by X-ray crystallography or NMR spectroscopy.

defined as:

$$cRMS(X,Y) = \min_{T} \sqrt{\frac{1}{n}\sum_{i=1}^{n}(||\mathbf{x}_i - T\mathbf{y}_i||^2)} \tag{4}$$

where $\| \cdot \|$ is the Euclidean L_2-norm and T is a rigid body transformation (rotation and translation). A closed form solution for T yelds the optimal tranformation.

- **Distance root mean square deviation**: Another common RMS shape similarity measure is based on comparing intra-molecular distances matrices, i.e., the matrix of distances between all C_α atoms in each structure. For a point X, this matrix is defined as

$$d_{ij}^X = \| \mathbf{x}_i - \mathbf{x}_j \| \tag{5}$$

The distance matrix RMS deviation (**dRMS**) of X and Y is then defined as:

$$dRMS(X,Y) = \sqrt{\frac{1}{n(n-1)}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(d_{ij}^X - d_{ij}^Y)^2} \tag{6}$$

Remark 4.2.1. *The cRMS measure is used after optimal superposition of equivalent positions of two structures, while in the dRMS measure, by using the internal distances, the need to optimally align the two conformations is removed. However the computation time of the dRMS measure becomes quadratic in the length of the correspondence between the two proteins.*

Given the matrix D of all pairwise distances within the molecule, where the distances are defined as in formula (5), it is possible to compute a binary matrix C, called *contact map*, for the protein as a symmetric $n \times n$ array such that:

$$C(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} 1 & \text{if } d_{ij}^X < t; \\ 0 & otherwise. \end{cases}$$

where t is a given threshold value (a common value is $t = 7\mathring{A}$). Thus, there exists a contact between residues $\mathbf{x}_i$ and $\mathbf{x}_j$ if and only if they are within a given distance t from one another in the protein structure.

The contact map of a protein is a particularly useful representation of protein structure that provides also useful information about the protein's secondary structure [54]. For example, cluster of contacts represent certain secondary structures: α-helices appear as bands along the main diagonal since they involve contact between amino acids and its four successors; β-sheets are thick bands parallel or anti-parallel to the main diagonal.

The overlap of two contact maps, as defined by the number of contacts between equivalent residues in two proteins that are simultaneously present in both structures, can be used as a measure of similarity between two protein structures ([41] and references in [70]).

Remark 4.2.2. *As we can see in [41], the method of contact map overlap is very useful when we have a low identity sequence. Indeed, contact map overlap between sequentially unrelated proteins is clearly distinguishable from the random overlap and quite close to the overlap between close sequential homologues. This makes this measure of structural similarity different from the cRMS between core residues, which was shown to increase rapidly with changes in sequence [73, 74].*

4.2.3 Universal Similarity Metric on contact maps

Recently, other methods to compare protein structures based on the so called Universal Similarity Metric (USM) have been shown.

It is the quantity defined by Li and Vitany in [76, 24] and described in section 2.1.1 (there named Normalized Information Distance):

$$d(x,y) = \frac{\max\{K(x|y^*), K(y|x^*)\}}{\max\{K(x), K(y)\}} \tag{7}$$

The value $K(x)$ is approximated by the size (i.e. number of bytes) of the compressed string $zip(x)$. (in particular the compression algorithm used was

Linux's compress version 4.2.4. Other compression algorithms were tested without significant changes.)

As the contact map method previously described, USM have no need for the optimal structural alignment.

In [70] the authors measure similarity between protein structures in the following way:

- choose a protein dataset
- extract from each pdb file the first chain[3] (if other than chain A is used from the PDB files this is shown in the name of protein with the letter of the chain, for example 1babB indicates that has been taken the chain B)
- produce a contact map for each of the pdb files in the dataset (the contact maps used have a distance threshold of 6.5 Å and distances are measured from C_α atoms)
- for each pair of protein contact maps c_1, c_2, compute $d(c_1, c_2)$ using the d in equation (7) to obtain the similarity distance between them. Store all inter-distances in a matrix.
- use an off-the-shelf software to cluster together proteins based on the inter-distances matrix (*http://www2.biology.ualberta.ca/jbrzusto/cluster.php*)

The results of the clustering are represented by a tree.

The contact map used by Krasnogor and Pelta is written in a file of two columns: one for the position of the amino acid i and the second for the position of the amino acid j with a distance from i (distance between the C_α atoms) under the 6.5 Å.

We can see below an example of these kind of files.

[3] we remember that a protein can be composed by more than one polipeptide chain: the single chains are usually considered as a single unit.

$$
\begin{array}{cc}
0 & 2 \\
0 & 3 \\
0 & 4 \\
1 & 3 \\
1 & 4 \\
1 & 5 \\
2 & 4 \\
2 & 5 \\
2 & 6 \\
\vdots & \vdots
\end{array}
$$

From their results, the USM distance seems to be capable of capturing protein similarities that encompass a variety of other, more euristic, criteria in a fully automated way [70]. Nevertheless, the results are considered controversial in [96], where the authors use a much larger and representative protein dataset than Krasnogor-Pelta datasets. The dataset is that used by Sierk and Pearson to evaluate seven protein structure comparison methods and two protein sequence comparison methods [98].

The data are composed by 2771 proteins, subset of CATH domains, and 86 of these are prototype proteins belonging to 86 different domain (according to CATH classification). Of the 2771 proteins, only 1120 belong to the 86 families of the 86 prototypes (so there is a maximum of 1120 correct hits).

These 86 proteins will be used from [38] too using a distance based on the compressor algorithms as well, and the results are comparable with those obtained with the alignment algorithms.

4.3 From primary sequence to structure

The methods that now we want to analyze can be seen as a step towards answering the folding problem: given the amino acid composition of a protein, how may one predict its folding type?

So far the rules bringing a protein to fold in its own ways are not clear. The idea that from the primary sequence it is possible to detect the principles that govern the folding of the protein chain is based on the thermodynamic hypoteis or Anfinsen Principle. This hypotesis states that "the three-dimensional structure of a native protein in its normal physiological milieu (solvent, pH, ionic strength, presence of other components such as metal ions or prosthetic groups, temperature, etc.) is the one in which the Gibbs free energy of the whole system is lowest; that is the native conformation is determined by the totality of interatomic interactions and hence by the amino acid sequence, in a given enviroment" [2].

Mutation and natural selection allowed a high degree of freedom during the evolution of species, or during accidental mutation, but a limited number of residues, destined to become involved in the internal, hydrophobic core of proteins, must be carefully conserved, or replaced with other residues with a close similarity in bulk and hydrophobicity.

These different mutations may sometimes bring to very different sequences still similar in strucure, and so in functions (homologous sequences). In more similar sequences it seems that even amino acid composition alone may be useful to assign a protein to a certain structural class, while, in this case of low identity sequences, it is evident that the regolatory rules are still more hidden. So, what are a sequence features regulating a protein folding in a precise way?

The primary structure is currently publicly known for hundreds of thousands of proteins, while the secondary and tertiary structure is known for a relatively small number of proteins (in the Protein Data Bank (PDB) [6] currently there are 49760, out of which only a small portion have correct secondary structure and tertiary structure information, while the SWISS-PROT database [50] store 359942 protein sequences). Experimental methods for the discovery of secondary and tertiary structure such as X-ray crystallography and nuclear magnetic resonance spectroscopy are time consuming, labor expensive, and cannot be applied to some proteins. Computational methods

perform prediction of the tertiary structure with an intermediate step of predictiong the secondary structure.

Computational methods for the prediction of secondary structure from the primary sequence aim to close the existing gap between the number of known primary sequences and higher structures.

Predictive methods based on probabilistic methods (such as Neural Network or HMM [34, 35], for example) use the informations in the amino acid sequence, coded as a symbolic string, to predict for *each* amino acid the possible secondary structure conformation. The tertiary structure prediction problem can be seen as a problem of *mapping*, i.e. the problem to try the mapping rules between the string representing the protein sequence and the symbolic string representing the structural feature that must be predicted.

The methods that will be shown now, instead, are more concentrated on trying to cluster proteins belonging to different classes, usually according the subdivision of SCOP dataset, and in case, to get from the good clusterization the rules that are typical for a group rather than another one.

In the next sections for each method we explain how it works and which kind of datasets are used. The analysis of these methods and of their results will be given in section 4.6.

4.3.1 Mahalanobis Distance and Coupled Method

In the years 80-90 the principal methods used in the prediction of structural classes for protein sequences worked on the aminoacid composition. In [22] the authors first, take into account the correlative effect among different amino acids and measure the similarity between two proteins with the Mahalanobis distance rather than the ordinary intutitive geometric distances, such as the Minkowski's distance or the Euclidean distance.

In the Mahalanobis Distance, indeed, the correlative effect among different amino acids can be automatically incorporated by the correlation matrix.

Among all the information in an amino acid sequence, it is decided to encode the amino acid sequence alone: so a protein molecule can be represented

by a vector or point in a 20-dimensional space [22].

Owing to the normalization of amino acid components, only 19 are indipendent. Therefore, the Mahalanobis distance based on the 20-dimensional amino acid composition space must be divergent and meaningless. To overcome the divergence difficulty, the Mahalanobis distance is defined in a 20-1=19-dimensional space. An invariance theorem given in [20] states that the values of the Mahalanobis distance will remain the same regardless of which one of the 20 components is left out for forming the reduced 19-dimensional space.

Suppose the 20 amino acids are alphabetically ordered according to their single-letter code.

A kth protein in a given protein set of cardinality N can be rappresented by

$$\mathbf{X}_k = \begin{bmatrix} x_{k,1} \\ x_{k,2} \\ \vdots \\ x_{k,19} \end{bmatrix} \tag{8}$$

with $k = 1, \ldots, N$ where $x_{k,i}$ is the frequency of amino acid i in the sequence k.

The mean of the protein set concerned is defined by

$$\mu_{\mathbf{X}} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_{19} \end{bmatrix} \tag{9}$$

where

$$\mu_i = \frac{1}{N} \sum_{k=1}^{N} x_{k,i} \tag{10}$$

$(i = 1, \ldots, 19)$.

When the N proteins in (9) are all α proteins, μ_X becomes the mean of α protein set, denoted by μ_{X_α}. Similarly, it is possible to define the mean of

the other protein sets β, $\alpha+\beta$ and α/β, when the N proteins in equation (9) are all β or $\alpha+\beta$ or α/β proteins respectively.

The Mahalanobis Distance between two proteins take into account the correlations between them with the use of the 19×19 covariance matrix:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_{1,1} & \sigma_{1,2} & \dots \sigma_{1,19} \\ \sigma_{2,1} & \sigma_{2,2} & \dots \sigma_{2,19} \\ \vdots & \vdots & \ddots \vdots \\ \sigma_{19,1} & \sigma_{19,2} & \dots \sigma_{19,19} \end{bmatrix} \tag{11}$$

where

$$\sigma_{i,j} = \sum_{k=1}^{N} \frac{1}{N-1}[x_{k,i} - \mu_i][x_{k,j} - \mu_j](i,j = 1,2,\dots,19) \tag{12}$$

Suppose now that $\mathbf{X}$ is a protein whose folding type is to be predicted. It can be either one of the N proteins in equation (8) or a protein outside of them.

The Mahalanobis Distance, $D_M^2(\mathbf{X}, \mu_X)$, between the mean μ_X defined in equation (9) and $\mathbf{X}$ in the 19-D space is given by:

$$D_M^2(\mathbf{X}, \mu_X) = (\mathbf{X} - \mu_X)^t \mathbf{\Sigma}^{-1} (\mathbf{X} - \mu_X)$$

where $\mathbf{\Sigma}^{-1}$ is the inverse matrix of the covariance matrix.

When N in equations (10) and (12) is equals to the number of all α proteins or all β or $\alpha+\beta$ or α/β proteins, then the covariance matrix becomes the covariance matrix of each protein set, denoted by Σ_ξ with $\xi \in FoldSet = \{\alpha, \beta, \alpha+\beta, \alpha/\beta\}$

In this case, we denote the distance between a sequence and a specific group with

$$D_M^2(\mathbf{X}, \mathbf{X}^\xi) = (\mathbf{X} - \mu_\xi)^{\mathbf{t}} \mathbf{\Sigma}_\xi^{-\mathbf{1}} (\mathbf{X} - \mu_\xi) \tag{13}$$

When $D_M^2(\mathbf{X}, \mathbf{X}^\xi)$ is smaller, meaning that the protein $\mathbf{X}$ is closer to the ξ protein set, and hence the likelihood of it belonging to the ξ folding type

is higher. Thus, the protein $\mathbf{X}$ will be predicted to be the folding type for which D_M^2 has the least value (Least Mahalanobis algorithm).

Incorporation of the component-coupled effects by introducing Mahalanobis distance and other advanced geometry distances was one big step forward in this area that significantly improved the prediction quality. However, the power of the least Mahalanobis distance algorithm was manifested only when the training subset sizes N_ξ, i.e. the number of proteins in the subsets of the training dataset, were the same or approximately the same; otherwise, poor predictions would result.

When the subset sizes are different, as shown in [21], it is like better to use the *Mahalanobis Discriminant Factor*, defined as following:

$$F(\mathbf{X}, \mathbf{X}^\xi) = D_M^2(\mathbf{X}, \mathbf{X}^\xi) + \log \prod_i^{20} \lambda_i^\xi - 2 \log \Psi_\xi + \Lambda \log(2\pi) \qquad (14)$$

where $\log \prod_i^{20} \lambda_i^\xi$ is the product of all positive eigenvalues of Σ_ξ, Ψ_ξ is the *a priori* probability of the subset ξ and Λ is the dimension of the amino acid composition space. The authors use a short version of this indicator, where the two last terms are not taken into account. Indeed, the last term is costant for all proteins and can be ignored. Since the prior probabilities Ψ_ξ are unknown, a common practice is to assume that they are equal. Then the term $2 \log \Psi_\xi$ can also be ignored and the equation (14) is reduced to

$$F(\mathbf{X}, \mathbf{X}^\xi) = D_M^2(\mathbf{X}, \mathbf{X}^\xi) + \log \prod_i^{20} \lambda_i^\xi \qquad (15)$$

Respect to previous distances, such as Minkowsky or Euclidean distances, the Covariant Discriminant Algorithm is not based on a distance scale but on a function that has incorporated both the component coupled effect and the subset size factor [19].

The use of the Mahalanobis Discriminant is also named *Coupled Method*. (In the following sections we will use this name for indicating this method). The Mahalanobis distance algorithm can be seen as an approximation of the component-coupled algorithm, and sometimes equation (13) is named

1st-Order Component-Coupled Method, while for equation (15) it is used the *2nd-Order Component-Coupled Algorithm*. When the matrix Σ_ξ is equal to the identity matrix, the mahalanobis distance becomes the Euclidea distance also named *0th-Order Component-Coupled Algorithm* [18].

Different ways of using this Mahalanobis Discriminant Function are related to the different ways to code the protein sequence. In the two next sections we will see other codes that take into account other features of the proteins, not only the amino acid composition.

Datasets used in [22] and [21] We give a name to those datasets that will be used by more than two authors. The dataset that are used only in an article are named with the number of proteins that are in the dataset.

In [22] the Mahalanobis distance is used on a dataset originally used in [86]:

Nakashima_Dataset: a dataset of 131 proteins choosen from the original dataset of 135 proteins in [86], where the irregular folding type proteins have been left out because their number is only four, too small to have any statistical significance. No informations on the livel of homology is given.

In [21] the Coupled Method is used mainly on five datasets:

138 a dataset of 138 protein domains extracted from SCOP database [85]. 36 of them are all-α domains, 29 all-β domains, 32 α/β domains and 41 $\alpha+\beta$ domains.

253 a dataset of 253 protein domains extracted from SCOP dataset with 63 all-α domains, 58 all-β domains, 61 α/β domains and 71 $\alpha+\beta$ domains

510 a dataset of 510 protein domains extracted from SCOP dataset with 109 all-α domains, 130 all-β domains, 135 α/β domains and 136 $\alpha+\beta$ domains

Chou_Dataset: a dataset of 359 higly homologous protein domains extracted from SCOP dataset with 82 all-α domains, 85 all-β domains, 99 α/β domains and 93 $\alpha+\beta$ domains

Moreover, for testing the prediction among seven structural classes (in addiction to the usually four, are considered also the μ (multi), σ (small) and ρ (peptides) domains) is used a dataset of 2438 protein domains extracted from SCOP dataset with 398 all-α domains, 704 all-β domains, 509 α/β domains, 608 $\alpha+\beta$ domains, 46 multi μ domains, 158 small protein σ domains and 20 peptide ρ domains.

No information on the level of homology is given for these datasets, but for the Chou_Dataset in [57] are found some homologous sequences (that would lead to overestimate the predictive accuracy of jacknife test).

4.3.2 Auto-correlation function-based approach (ACF)

Prediction of the protein structural classes is usually performed as a two step procedure. First, sequences of different length are represented by a fixed length feature vector and next the feature values are fed into a classification algorithm.

In the previous section it was observed that the structural class of a protein is related to its amino acid composition and its prediction improves by including the coupling effect among different amino acid components. Although the amino acid composition is very suitable to calculate, the full information contained in the primary sequence is reduced considerably.

Bu et al. in [7] developed the Auto-correlation function-based approach (ACF) to deal with this problem: for coding the protein sequence, instead of the amino acid composition, it here are taken into account various physicochemical and biochemical properties of amino acids through the use of amino acid index. An amino acid index is a set of 20 numerical values representing any of the different physicochemical properties of the 20 amino acid. In the article it is used the index of Oobatake and Ooi that is an indicator of the

average nonbonded energy per residue (see references in [7]).

In this method the protein is represented by a vector $\mathbf{X} = (r_1, r_2, \ldots, r_m)^t$ where r_i are the autocorrelation functions between amino acid indexes. For calculating the auto-correlation functions, first each residue in the primary sequence is replaced by its amino-acid index, so that each protein sequence becomes a numerical sequence $h_1, \ldots, h_N$, where h_i is the amino acid index for the ith residue and N is the length of the protein sequence.

The auto correlation r_n is so defined:

$$r_n = \frac{1}{N-n} \sum_{i=1}^{N-n} h_i h_{i+n}$$

with $n = 1, 2, \ldots, m$ and m is the number of the auto-correlation functions used. The optimal value of m is dependent on the kind of amino acid index used; in the case of amino acid index of Oobatake and Ooi used in this article, the appropriate value for m is 30.

Datasets used in [7] The datasets used in [7] are:

138 the same dataset described in the previous section (used in [21])

510 the same dataset described in the previous section (used in [21])

Chou_Dataset: the same dataset of 359 proteins described in the previous section and used also in [21]

4.3.3 Pseudo Amino Acid Composition

In other articles this idea of using the autocorrelation function in the code of the original aminoacid sequence, is taken into account in order to improve the prediction quality for protein structural classification by effectively incorporating some kind of sequence-order effects.

Indeed, when we use the conventional amino acid composition to represent the sample of a protein, all its sequence order and length effects are lost.

To include these effects, the concept of *pseudo amino acid composition* was introduced in [20].

The core of using pseudo amino acid composition is, on the one hand, to include the main feature of amino acid composition, but, on the other hand, to include information beyond amino acid composition.

In [20], a protein chain of L amino acid residues: $R_1, R_2, \ldots, R_L$, is rappresented by a $20 + \lambda$ dimensional vector, where the first twenty components are the aminoacid frequencies and the others are autocorrelation functions that reflect the sequence order correlation between all the ith most contiguous residues.

In [20] are defined the following order-correlated factors:

$$\begin{cases} \theta_1 = \frac{1}{L-1} \sum_{i=1}^{L-1} \Theta(R_i, R_{i+1}) \\ \theta_2 = \frac{1}{L-2} \sum_{i=1}^{L-2} \Theta(R_i, R_{i+2}) \\ \theta_3 = \frac{1}{L-3} \sum_{i=1}^{L-3} \Theta(R_i, R_{i+3}) \\ \qquad\qquad \vdots \\ \theta_\lambda = \frac{1}{L-\lambda} \sum_{i=1}^{L-\lambda} \Theta(R_i, R_{i+\lambda}) \quad (\lambda < L) \end{cases} \tag{16}$$

where θ_i is the ith correlation factor that reflect the sequence order correlation between all the ith most contiguous residues and the correlation function in (16) is given by

$$\Theta(R_i, R_j) = \frac{1}{3}\{[H_1(R_j) - H_1(R_i)]^2 + [H_2(R_j) - H_2(R_i)]^2 + [M(R_j) - M(R_i)]^2$$

where H_1, H_2, M are respectively the hydrophobicity value, hydrophilicity value and side-chain mass of the amino acid. So, instead of using a 20-dimensional vector to represent a protein, it is used a $20 + \lambda$-D vector:

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{20} \\ x_{20+1} \\ \vdots \\ x_{20+\lambda} \end{bmatrix}$$

where

$$x_u = \begin{cases} \frac{f_u}{\sum_{i=1}^{20} f_i + \omega \sum_{j=1}^{\lambda} \theta_j} & (1 \leqslant u \leqslant 20) \\ \frac{\omega \theta_{u-20}}{\sum_{i=1}^{20} f_i + \omega \sum_{j=1}^{\lambda} \theta_j} & (20+1 \leqslant u \leqslant 20+\lambda) \end{cases}$$

where f_i is the normalized occurrence frequency of the 20 amino acids in the protein $\mathbf{X}$, θ_j is the j-th sequence correlation factor and ω is the weight factor for the sequence order effect (in the article $\omega = 0.05$).

Although the incorporation of correlation functions in the pseudo amino acid seems to give good results in subcellular location problems [37], in the structural class problem is not so used.

In general, to further improve the prediction quality, the pseudo amino acid composition should be optimized by reducing the number of its additional components and increasing the sequence-order information in the remaining components.

This is gained in [109] introducing the *complexity measure factor* into the pseudo amino acid composition. In this way a protein $\mathbf{X}$ can be expressed by a 21-dimensional vector, where the first 20 components are the frequency occcurrences of the amino acids in the protein, the last component is the complexity value calculated according the lz procedure (see section1.2). On this code of proteins it is applied the usual methodology of the covariant discriminant algorithm [21].

Datasets used in [109] In [109] it is used a single dataset originally created by Chou in [18]:

204 non homology-dataset : a dataset of 204 *entire protein chains* (no domains) extracted from the SCOP dataset. It consists of 52 all α, 61 all-β, 45 α/β and 46 $\alpha+\beta$.

We recall that the SCOP dataset is based on the domain proteins, but the authors in [18] constructed this dataset such that any protein in the new dataset must, as a *whole*, clearly and unambiguously belong to one of the four structural classes. In this dataset, the level of homolgy is under the 30%.

4.3.4 Discriminant Analysis with peptides

We will see in the results section 4.6 that the use of codes that try to capture also the residue order along sequence gives an improvement. The other methods here shown, are based on the comparison of subsequence distributions. Indeed, the set of subsequence distributions of a protein domain involves the residue order along the sequence; therefore they incorporate more information of the sequence than its amino acid composition does.

In [78] these subsequences (or *peptidi*) are chosen with a stepwise discriminant analysis in multivariate statistics. Starting from the set of amino acid $T(1)$, the first step constructs a subset of $T(1)$, denoted by $T_0(1)$ with the use of a discriminant analysis where the variables are the frequency vectors of the amino acids in the different groups involved in the attribution (these frequency variables are denoted by the vector $X(i)$). The set of di-peptides $T(2)$ is constructed from $T_0(1)$ add each of the 20 elements in $T(1)$ in front and at the end of the elements in $T_0(1)$. Before applying another discriminant analysis on the new set, obtaining a subset $T_0(2)$ with only "significant" di-peptides, some variable are left out if their mean on each set is lower than a threshold. The discriminant results are checked while polypeptides are taken into account: step by step all the variables $X(1), X(2), \ldots, X(i+1)$ chosen by the discriminant procedure are put togheter for performing another stepwise discriminant analysis with a discriminant result $R(i+1)$. If none

of $X(i+1)$ enters the model for discrimination, or if $R(i+1) < R(i)$ than the process stops. The variables obtained from $X(1), X(2), \ldots, X(i+1)$ by stepwise discrimination are the desired variables in final prediction and $R(i)$ is the highest predictive accuracy. These variables can be further on used for the classification with other datasets.

The results of [78] are compared with those obtained on the same datasets with the coupled method, and their accuracy is considerably better.

Nonetheless, an observation is necessary: as usual the accuracy of the method is valueted by the jackknife and resubstitution test. For what concerns the jackknife procedure, the polypeptides chosen in the stepwise discrimination are used in the jackknife manner but the polypeptides were computed using the entire dataset, which means that information about the tested protein was used to perform its prediction. So the results obtained require more detailed attention (see [71]).

Datasets used in [78] In [78] three datasets are taken into account:

PDB40-b: 1054 proteins extracted from SCOP [85] with the pairwise sequence identity less than 40%. The proteins belong to the first four major structural classes: 220 are in all-α class, 309 are in all-β class, 285 are in α/β class and 240 are in $\alpha+\beta$ class.

758 a datset of 758 proteins (in the article named PDB40-j) extracted from SCOP with the pairwise sequence identity less than 40%. 162 are in all-α class, 209 re in all-β class, 222 are in α/β class and 165 are in $\alpha+\beta$ class.

372 : a dataset of 372 sequences (in the article named PDB40-j1) obtained excluding from PDB40-j the 386 identical sequences between PDB40-b and PDB40-j.

Chou_Dataset the same dataset of 359 proteins also used in [21]

4.3.5 Discrepancy measure on polypeptides

In [57] given a protein sequence S with L residues, it is calculated the subsequence distribution of S with the subsequence length equal to ℓ, and on this distribution is calculated a measure of information discrepancy.

The number of all different contiguous sequences formed from an alphabet $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$ with length ℓ, is $m(\ell)$ and it is equal to m^ℓ.

Let be n_i^ℓ the number of the i contiguous subsequences in S of length ℓ. The total number of subsequences of length ℓ in S is $L-\ell+1$ for each $\ell \leqslant L$. Define $p_i^\ell = \frac{n_i^\ell}{L-\ell+1}$, we obtain the distribution:

$$U_S^l = (p_1^l, p_2^l, \ldots, p_{m(l)}^l)^t$$

Any sequence can be uniquely recognized by increasing the length of the subsequence. Given a set of distribution of s sequences, the FDOD (Function of Degree of Disagreement) measure is defined as:

$$B(U_{S_1}^l, U_{S_2}^l, \ldots, U_{S_s}^l) = \sum_{k=1}^{s} \sum_{i=1}^{m(l)} p_{ik}^l \log \frac{p_{ik}^l}{\sum_{k=1}^{s} \frac{p_{ik}^l}{s}} \tag{17}$$

$$B_k(U_{S_1}^l, U_{S_2}^l, \ldots, U_{S_s}^l) = \sum_{i=1}^{m(l)} p_{ik}^l \log \frac{p_{ik}^l}{\sum_{k=1}^{s} \frac{p_{ik}^l}{s}} \tag{18}$$

where $0 \log 0 = 0$ and $0 \log \frac{0}{0} = 0$. $B(U_{S_1}^l, U_{S_2}^l, \ldots, U_{S_s}^l)$ denotes a measurement of the discrepancy among s sequences; while $B_k(U_{S_1}^l, U_{S_2}^l, \ldots, U_{S_s}^l)$ denotes a measurements of the discrepancy between the $k-$ sequence and all other sequences in the group.

As usual, if we suppose that a set T is the union of four subsets of sequences: $T = T^\alpha \cup T^\beta \cup T^{\alpha/\beta} \cup T^{\alpha+\beta}$ and denote the discrepancy of T^α and X by B_X^α (similarly for the other groups), the query protein X will be assigned to class R when $B_X^R = min\{B_X^\alpha, B_X^\beta, B_X^{\alpha/\beta}, B_X^{\alpha+\beta}\}$ $(R = \alpha, \beta, \alpha/\beta, \alpha+\beta)$

The better results, are achieved with $\ell = 3$. This is also the value considered in the results shown in our tables.

In [64] the authors show that the high prediction accuracy obtained with the FDOD measure on a low homology dataset is an artifact of improper implementation that falsifies the results obtained with the jaccknife test; indeed the average distribution used in the denominator of (18) is pre-calculated using the whole dataset. Following a right implementation of the method, the results considerably decrease (see table 1 in [64])

Datasets Used in [57] The mainly chosen datasets by [57] are focused on the value of sequence redundancy The proteins taken from the SCOP dataset are filtered according to percentage sequence redundancy with a HMM library and genome assignments server named Superfamily [44]. In [57] the datasets are:

Chou_Dataset: the dataset of 359 proteins described above

T30-1401_Dataset: a dataset of 1401 proteins (domains) extracted from the SCOP dataset, derived from the Superfamily sequence list under the limits of 30% sequence redundancy. This dataset consists of 298 all-α proteins (domains), 341 all-*beta* proteins (domains), 438 α/β proteins (domains) and 324 $\alpha + \beta$ proteins (domains).

1530 a dataset of 1530 protein domains (in the article named T10-1530) with no more than 10% redundancy.

1564 a dataset of 1564 protein domains (in the article named T10-1530) with no more than 20% redundancy.

3998 a dataset of 3998 protein domains (in the article named T90-3998) with no more than 90% redundancy.

4.3.6 Compression-based distance measures to protein sequence classification

In the last years, other authors use the compression based distance measures for the classification of protein sequences. Respect to the previous methods,

the data on which these methods work is the amino acid sequence itself, the **primary** sequences without coding other features.

These techniques are also indipendent from the resolution of the sequence, i.e. they do not involve segments of fixed length.

Combination of the CBM and the BLAST score

In [67] the authors make a combination of the BLAST score and compression-based measures in the following formula:

$$F(X,Y) = \left(1 - \frac{S(X,Y)}{S(X,X)}\right) CBM(X,Y) \tag{19}$$

where $S(X,Y)$ is a BLAST score computed between a query X and a subject Y, $S(X,X)$ is the BLAST score of the query compared with itself. With CBM the authors denote the compression based distance measures (the NCD measure of section 2.1.2):

$$CBM(X,Y) = \frac{C(XY) - min\{C(X), C(Y)\}}{max\{C(X), C(Y)\}} \tag{20}$$

where C denotes a compressor such as the Lempel-Ziv with the exhaustive parsing [72] or the PPMZ compression algorithms. The term in parenthesis of (19) is used to transform the BLAST score into a normalized distance measure that ranges between zero and one.

The performance of this CBM+BLAST method is compared with other alignment algorithms, in particular they note that it is close to that of the Smith-Waterman algorithm (in some cases, even slightly better).

For this comparison the authors use nearest neighbour or support vector machine classification schemes. In the SVM method, a sequence X is represented by a feature vector $F_X = f_{x1}, f_{x2}, \ldots, f_{xn}$, n is the total number of proteins in the training set and f_{xi} is a similarity/distance score, such as BLAST score, the Smith-Waterman score or a CBM, between sequence X and the $i-$th sequence in the training set.

Datasets used in [67] The datasets used are four:

Dataset1: consisting of 4352 protein domain sequences selected from the SCOP database, belonging to 54 different superfamilies. This dataset is an example of distant protein similarity

Dataset 2: 131 sequences of the 3-phosphoglycerate kinase obtained from 15 archean, 83 bacterial and 33 eukaryotic species. This dataset is an example of evolutionarily related sequences. (similar proteins that make the same biological function. Example of homology sequences)

Dataset 3: artificial sequences designed to study the behaviour of CBMs. Each of the 4352 sequences in Dataset 1 were random shuffled and compared with its original counterpart

Dataset 4: sequences of high and low complexity.

The results on the dataset 1, 2 are comparable to those obtained with the Smith-Waterman algorithm. From the dataset 3 we can obtained that a reshuffling on the domains has a smaller effect on the compression distance than on the BLAST score. But there is a problem: similarities are more difficult to detect on short sequences than on long ones, so the method is strongly dependent on the sequence length.

On the last dataset, they observed that the distribution of CBM values on low and high complexity proteins were not markedly different (see [67])

The goal of the authors is to compare their measure with the results obtained with the alignment-based method. The alignment-based comparison of biological sequences plays a fundamental role in most areas of computational genomics. Although the most popular alignment-algorthm is BLAST [1], this doesn't achieve the best alignment between two or more sequence, as an exhaustive algorithm such as the Smith and Waterman algorithm does [101]. But, considered the computational complexity of the Smith Algorithm, it is preferred the use of an heuristic algorithm such as BLAST.

The results in [67] show that the method (19) is comparable with the Smith-Waterman Algorithm and, sometimes, better than BLAST alone.

Compressor Algorithms

An exhaustive use of the measure (20) and other similar with different compression algorithms, is given in [38].

They used twenty different compression algorithms on five different protein datasets with three different metrics (all on the basis of the CBM); their accurracy is valueted with the ROC analysis and the corrisponding AUC value, while in order to assess the performance of compression-based classification, via UPGMA or NJ, they computed the F-measure.

The comparison of these methods is done with the alignment method BLAST

Datasets used in [38] In [38] mainly two dataset are used. For each dataset different ways to code the proteins are chosen, obtaining in total five datasets:

CK_Dataset the Chew-Kedem dataset, already studied in [70], consists of 36 protein domains drawn from the PDB entries of three classes (alpha-beta, mainly alpha, mainly beta)

1. **CK-36-PDB:** Chew-Kedem dataset of 36 proteins domains, amino acid sequences in FASTA format
2. **CK-36-REL:** Chew-Kedem dataset of 36 proteins domains, complete TOPS strings with contact map
3. **CK-36 SEQ:** Chew-Kedem dataset of 36 proteins domains, TOPS strings of secondary structure elements

SP_Dataset the Sierk-Pearson dataset [98], consists of a non redundant subset of 2771 protein families and 86 non-homologous protein families from the CATH proteins domain database [91].

4. **SP-86-PDB:** Sierk Pearson dataset of 86 protein domains, amino acid sequences.
5. **SP-86-ATOM:** Sierk-Pearson dataset of 86 protein domains, ATOM lines from PDB entries

4.4 Our (very) preliminary experiments: LZ distance and n-gram distance

We repeated the experiment on one of the datasets used by Krasnogor and Pelta in [70]; in particular we take the Chew-Kedem dataset, where there are 36 proteins.

We recall that the authors in [70] used the USM distance on proteins rappresented by files of contact couples (see section 4.2.3).

The distances used in our experiments are the two distances that we have already used in the heart experiments: the LZ-distance and the n-gram distance (go to the sections 2.1.3 and 2.2.6 for the formulas).

The contact maps are obtained from the first chain of the PDB files: two amino acids i and j are in contact when the distance between their C_α atoms is below 6.5 Å.

The contact map in [70] was coded as a file of two column where the first column is the position of amino acid i and in the second column the position of the amino acid j with which i is in contact.

We have used this kind of code for the contact map with our distances, but we have also constructed the whole contat map reading for lines, along the bias or choosing a window of reading.

All the matrices obtained with different methods and different value of n (for the n-gram methods) are evaluated by the three attribution methods already explained in section 3.2.2: the mean method, the nearest neighboor and the vertical voting.

We have also used these attribution methods on the matrix obtained by

Krasnogor and Pelta using the USM distance.

The results are summarized in table 4.2: for the n-gram method we show the value of n for which we obtained the best results. As you can see from the table, our methods, on this dataset and with these ways of attribution, reached better results than those with the USM method; the best result (91%) is achieved with the method of n-gram, with the formula with $\alpha = 2$, an $n = 9 - 10$, and reading the contact map for lines. With the same reading of the contact map used in [70] (named CMP in the table), we obtained higher percentages than those of Krasnogor and Pelta, for each attribution method.

In order to asses the performance of our methods on the classification, following [38] we have applied the UPGMA clustering algorithm on our distances, and then we have computed the F-measure by using the clustering solution and the gold standard division of proteins in groups according to CATH class (we have used the same procedures used by [38]. See the reference in the article for the suplementary material web page).

In this case, the LZ-distance and n-gram distance are used on the amino acid sequences, and the results are compared with those shown in [38] by the compressors, and also with the alignment methods (see table 4.3).

Among all the results of Ferragina at al. we show only the better results for each kind of compressors. Better results compared to F-measure are obtained only for the PPMd algorithm (an adaptive statistical data compression technique based on context modeling and prediction): all other compressor algorithms have values lower than our results. As we will observe in one of the following sections (see section 4.6), the alignment algorithm performs very well on this dataset that, probably it contains homologous proteins.

An intersting dataset is the **SP-dataset** used also in [38] that contains 86 non homologous proteins. In this case we can observe an abrupt decrease in the performance of the alignment algorithms.

In [38] the compressor algorithms are used on this dataset with results that sometimes are slightly better than what we get with the allignment.

With our distances we obtain results comparable to those in [38]; in this case the LZ distance performs like the n-gram distances.

Moreover, in both the two datasets the best formula is the geometric formula (see tables 4.3 and 4.4).

4.5 Considerations on some of the previous methods

In this section we want to analyze differences and similarities between some of the methods shown in the previous section. The analysis is made on variables with normally gaussian distribution.

Rewriting some of the previous methods under the assumption that the variables have normally gaussian distribution, brings to the need of managing formulae with joint and conditional probability in it. For the sake of clearness we recall here in the forth coming the formulas relative to those involving two (both) gaussian variables probability.

4.5.1 Gaussian distributions

The general multivariate normal density is written as

$$p(X) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_x|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(X-\mu_x)^t\Sigma_x^{-1}(X-\mu_x)\right] \tag{21}$$

where X is a $d-$component column vector, μ_x is the $d-$component mean vector, Σ_x is the $d \times d$ covariance matrix, and, as usual, the Σ_x^{-1} is the inverse of Σ_x, $(X-\mu_x)^t$ is the transpose of $(X-\mu_x)$ and $|\Sigma_x|$ is the determinant of Σ_x.

Let :

$$\mu = E[X]$$

and

$$\Sigma = E[(X-\mu)(X-\mu)^t]$$

The same for $p(Y)$.

The joint probability is:

$$p(X,Y) = \frac{1}{(2\pi)^{\frac{d+d}{2}}|\Sigma_z|^{-\frac{1}{2}}} \exp\left[-\frac{1}{2}(Z-\mu_z)^t\Sigma_z^{-1}(Z-\mu_z)\right] \tag{22}$$

where $Z = (X,Y)^t$.

From these formulas we can get the conditional probability density:

$$\begin{aligned} p(X|Y=y) &= \frac{p(X,y)}{p(y)} \\ &= \frac{1}{\frac{(2\pi)^{\frac{d+d}{2}}|\Sigma_z|^{\frac{1}{2}}}{(2\pi)^{\frac{d}{2}}|\Sigma_y|^{\frac{1}{2}}}} \exp\left[-\frac{1}{2}\left((Z-\mu_z)^t\Sigma_z^{-1}(Z-\mu_z) - (y-\mu_y)^t\Sigma_y^{-1}(y-\mu_y)\right)\right] \\ &= \frac{1}{(2\pi)^{\frac{d}{2}}\frac{|\Sigma_z|^{\frac{1}{2}}}{|\Sigma_y|^{\frac{1}{2}}}} \exp\left[-\frac{1}{2}\left((Z-\mu_z)^t\Sigma_z^{-1}(Z-\mu_z) - (y-\mu_y)^t\Sigma_y^{-1}(y-\mu_y)\right)\right] \end{aligned}$$

The conditional probability can also be directly written as follows:

$$p(x|y) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_{x|y}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(X-\mu_{x|y})^t\Sigma_{x|y}^{-1}(X-\mu_{x|y})\right] \tag{23}$$

The mean and covariance of the conditional distribution of X given a realization of Y (that has normal distribution itself) are:

$$\Sigma_{x|y} = \Sigma_x - \Sigma_{xy}\Sigma_y^{-1}\Sigma_{yx} \tag{24}$$

$$\mu_{x|y} = \mu_x + \Sigma_{xy}\Sigma_y^{-1}(y-\mu_y) \tag{25}$$

This result can be obtained directly comparing the two equations (22) and (23) and using the properties of the gaussian distribution.

Here, following [58], we show an indirect proof.

Proof. Take a matrix A of dimension equal to $2d \times 2d$:

$$A = \left(\begin{array}{c|c} I & -\Sigma_{xy}\Sigma_y^{-1} \\ \hline 0 & I \end{array}\right)$$

so

$$A(Z-\mu_Z) = A\left(\begin{array}{c} X-\mu_x \\ \hline \\ Y-\mu_y \end{array}\right) = \left(\begin{array}{c} X-\mu_x-\Sigma_{xy}\Sigma_y^{-1}(Y-\mu_y) \\ \hline \\ Y-\mu_y \end{array}\right)$$

is jointly normal with covariance matrix $A\Sigma_z A^t$ given by:

$$\left(\begin{array}{c|c} I & -\Sigma_{xy}\Sigma_y^{-1} \\ \hline 0 & I \end{array}\right)\left(\begin{array}{c|c} \Sigma_x & \Sigma_{xy} \\ \hline \Sigma_{yx} & \Sigma_y \end{array}\right)\left(\begin{array}{c|c} I & 0 \\ \hline -\Sigma_{xy}\Sigma_y^{-1} & I \end{array}\right) = \left(\begin{array}{c|c} \Sigma_x-\Sigma_{xy}\Sigma_y^{-1}\Sigma_{yx} & 0 \\ \hline 0 & \Sigma_y \end{array}\right)$$

The variable $X-\mu_x-\Sigma_{xy}\Sigma_y^{-1}(Y-\mu_y)$ has normal distribution $N(0,\Sigma_x-\Sigma_{xy}\Sigma_y^{-1}\Sigma_{yx})$. Moreover, if we consider the two variables $X-\mu_x-\Sigma_{xy}\Sigma_y^{-1}(Y-\mu_y)$ and $Y-\mu_y$, they have zero covariance and then they are indipendent. This means that when $Y=y$ then conditional distribution of $X-\mu_x-\Sigma_{xy}\Sigma_y^{-1}(y-\mu_y)$ is the same as the unconditional distribution of $X-\mu_x-\Sigma_{xy}\Sigma_y^{-1}(Y-\mu_y)$. Since the term $\mu_x+\Sigma_{xy}\Sigma_y^{-1}(y-\mu_y)$ is a constant when $Y=y$, the variable X is distributed as $N(\mu_x+\Sigma_{xy}\Sigma_y^{-1}(y-\mu_y),\Sigma_x-\Sigma_{xy}\Sigma_y^{-1}\Sigma_{yx})$

□

4.5.2 Coupled Method vs. Bayes decision rule

We have seen that in the protein structural class prediction, the most used is a method based on Mahalanobis Distance, the Coupled Method.

As already observed in [116], the use of the coupled method for protein structural class predicition can be rewritten by the use of Bayes decision rule.

The Bayes' decision theory forms the basis of statistical *pattern recognition*. The theory is based on the assumption that the decision problem can be specified in probabilistic terms and that all of the relevant probability values are known.

Suppose there are M possible pattern classes $\xi_1,\xi_2,\ldots\xi_M$ and an arbitrary pattern $\mathbf{x}=x_1\ldots x_d$ belongs to class ξ_i with *a priori* probability

$P(\xi_i)$. With the *posterior probability* $P(\xi_i|\mathbf{x})$ or the class-conditional probability density $p(\mathbf{x}|\xi_i)$ and the prior probability $P(\xi_i)$ we can write Bayes' decision rule as:

$$P(\xi_i|\mathbf{x}) > P(\xi_j|\mathbf{x}) \ (i \neq j) \tag{26}$$

that is, the pattern class ξ_i with the higest posteriori probability is chosen as the assignment for $\mathbf{x}$.

Using the Bayes'theorem, it is possible to write:

$$P(\xi_i|\mathbf{x}) = \frac{P(\xi_i)P(\mathbf{x}|\xi_i)}{P(\mathbf{x})}$$

and the Bayes decision rule becomes:

$$P(\xi_i)P(\mathbf{x}|\xi_i)$$

where $P(\mathbf{x})$ has been eliminated since it does not depend on i. [4]

Usually, in pattern recognition problems, it is used a gaussian classifier that uses Bayes' decision theory where the class-conditional probability density $p(\mathbf{x}|\xi_i)$ is assumed to have a Gaussian distribution for each class ξ_i; let $P(\mathbf{x}|\xi_i) \sim N(\mu_i, \Sigma_i)$, [29] then:

$$P(\mathbf{x}|\xi_i) = \frac{1}{\sqrt{(2\pi)^d|\Sigma_i|}} \exp\{-\frac{1}{2}(\mathbf{x}-\mu_i)^t\Sigma_i^{-1}(\mathbf{x}-\mu_i)\} \tag{27}$$

where μ_i is the $d-$component mean vector and Σ_i is the $d \times d$ covariance matrix.

Let's take the logarithm of the decision function:

$$d_i(\mathbf{x}) = \ln[P(\xi_i)P(\mathbf{x}|\xi_i)] = \ln P(\xi_i) + \ln P(\mathbf{x}|\xi_i) \tag{28}$$

Substituting equation (27) in equation (28) yelds:

$$d_i(\mathbf{x}) = \ln P(\xi_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_i| - \frac{1}{2}[(\mathbf{x}-\mu_i)^t\Sigma_i^{-1}(\mathbf{x}-\mu_i)] \tag{29}$$

[4] It can be verified that the decision rule (26) minimizes the average of classification error probability

As shown in [116], multiplying by factor -2 both sides of equation (29), we get:

$$D_i(\mathbf{x}) = -2d_i(\mathbf{x}) = -2\ln P(\xi_i) + d\ln 2\pi + \ln|\Sigma_i| + [(\mathbf{x}-\mu_i)^t\Sigma_i^{-1}(\mathbf{x}-\mu_i)] \quad (30)$$

that, reorganizing the order of terms, becomes:

$$D_i(\mathbf{x}) = [(\mathbf{x}-\mu_i)^t\Sigma_i^{-1}(\mathbf{x}-\mu_i)] + \ln|\Sigma_i| - 2\ln P(\xi_i) + d\ln 2\pi \quad (31)$$

We see that, when we assume an equal distribution for all class ξ_i, the formula (31) is identical to the Discriminant Mahalanobis Factor, also named Coupled Method described in the previous section.

Therefore, the decision process uses the squared Mahalanobis distance and it is equivalent to the Coupled Method (in the Gaussian assumption) [107].

4.5.3 Information distances

All the methods that we have seen relate the similarity to the probability of two items belonging to the same cluster.

The idea of the Coupled method, estimating the $p(x|\xi_i)$, is to use a distance between a single item and the probability of the class (estimated with a group of items of which we know the class attribution). Other approaches, such as those correlated to the information distance, use the same concept of similarity but estimate the $p(x|y)$ where x and y are two different items instead of $p(x|\xi_i)$. The distances used in the compressor methods are *pair-wise* distances that try to estimate the similarity (or not) directly by the information that one item conveys about the other.

The quantity used in the theory of the information distance for estimating the shared features, as underlined in section 2.1, is the joint distribution $p(x, y)$; the ratio of this quantity from the indipendent hypotesis is an indicator of how much the two variables x and y are similar.

We consider the logaritmic form:

$$\begin{aligned} \log \frac{p(x,y)}{p(x)p(y)} &= \log \frac{p(x,y)}{p(y)} - \log p(x) \\ &= \log p(x|y) - \log p(x) \end{aligned} \tag{32}$$

Following [49], we can write (32) for gaussian distributions under two simplifying Gaussian assumptions:

1. $p(x,y)$ is gaussian in $\mathcal{R}^{2d}$
2. $p(z) = \int_x p(x,z)dx = \int_y p(z,y)dy$

(From the first assumption follows that $p(x)$ is also Gaussian in $\mathcal{R}^d$.)

Assuming that the data's mean is 0 (otherwise, we can substract it from the data), the quantities in equation (24) become:

$$\begin{aligned} \Sigma_{x|y} &= \Sigma_x - \Sigma_{xy}\Sigma_y^{-1}\Sigma_{yx} \\ \mu_{x|y} &= \Sigma_{xy}\Sigma_y^{-1}y \end{aligned} \tag{33}$$

and then, the equation (32) becomes:

$$\frac{1}{2}\left[\log \frac{|\Sigma_x|}{|\Sigma_{x|y}|} + x^t\Sigma_x^{-1}x - (x-My)^t\Sigma_{x|y}^{-1}(x-My)\right] \tag{34}$$

where, $M = \Sigma_{xy}\Sigma_y^{-1}$ and $\Sigma_{x|y}$ is given in (33)

This last formula, freed from multiplicative and additive constants, is the formulation of the coding similarity between two variables with gaussain distribution (GCS) and depends only on the data covariance matrix and the covariance between pairs from the same sources.

In [49] it is also proved that GCS, when the two points x and y are conditionally indipendent given the hidden source, is strongly related to Fisher Linear Discriminant (FLD) proving that the matrices employed in its computation are the within-class (Σ_x) and between-class (Σ_{xy}) scatter matrices involved in FLD. [5]

[5] we remeber that the Fisher-LDA considers maximizing the objective $J(w) = \frac{\mathbf{w}^t S_B \mathbf{w}}{\mathbf{w}^t S_W \mathbf{w}}$ where S_B is the "between classes scatter matrix" and S_W is the "within classes scatter matrix" as defined in [30]

Morever, when the points in all pairs are very close to each other, we can prove the convergence of the GCS to the Coupled Method (equation (15)). The theorem is an analogous of Therem 2 in [49], where the multiplicative and additive constants are not removed:

Theorem 4.5.1. *Let x be a point generated with probability $p(x)$ and y sampled from a small neighborhood of x. Denote $\Delta = \frac{(x-y)}{2}$. Assume that the covariance matrix $\Sigma_\Delta < \varepsilon\Sigma_x$, where $\varepsilon > 0$ and $A \leqslant B$ stands for "$B - A$ is a positive semi defined matrix". Then*

$$\textit{equation (34)} \rightarrow_{\varepsilon\rightarrow 0} -\frac{1}{2}[(x-y)(4\Sigma_\Delta)^{-1}(x-y) + \log|4\Sigma_\Delta|] \tag{35}$$

where the limit $g(x) \rightarrow f(x)$ means $\frac{g(x)}{f(x)} \rightarrow 1$

Proof. The proof of the theorem follow from that of Theorem 2 in [49]. Here are retrieved the main steps:

Denote with $\mu = \frac{x+y}{2}$; then we can rewrite x and y as following:

$$x = \mu + \Delta \text{ and } y = \mu - \Delta \tag{36}$$

First, we rewrite the covariance matrix for x (that for the initally assumption 2. is equal to the covariance matrix of y):

$$\begin{aligned}
\Sigma_y &= \Sigma_x = \frac{1}{2}E[xx^t] + \frac{1}{2}E[yy^t] = && (37)\\
&= \frac{1}{2}E[(\mu+\Delta)(\mu+\Delta)] + \frac{1}{2}E[(\mu-\Delta)(\mu-\Delta)] && (38)\\
&= \Sigma_\mu - \Sigma_\Delta && (39)
\end{aligned}$$

The Σ_{xy} becomes:

$$\begin{aligned}
\Sigma_{xy} &= E[xy^t] = E[(\mu+\Delta)(\mu-\Delta)] = \Sigma_\mu - \Sigma_\Delta && (40)\\
&= \Sigma_x - 2\Sigma_\Delta && (41)
\end{aligned}$$

Moreover:

$$\begin{aligned} M &= \Sigma_{xy}\Sigma_x^{-1} = (\Sigma_x - 2\Sigma_\Delta)\Sigma_x^{-1} && (42)\\ &= I - 2\Sigma_\Delta\Sigma_x^{-1} && (43)\\ \Sigma_{x|y} &= \Sigma_x - \Sigma_{xy}\Sigma_y^{-1}\Sigma xy = \Sigma_x - (\Sigma_x - 2\Sigma_\Delta)\Sigma_y^{-1}(\Sigma_x - 2\Sigma_\Delta) && (44)\\ &= \Sigma_x - (I - 2\Sigma_\Delta\Sigma_x^{-1})(\Sigma_x - 2\Sigma_\Delta) && (45)\\ &= 4\Sigma_\Delta - 4\Sigma_\Delta\Sigma_x^{-1}\Sigma_\Delta && (46) \end{aligned}$$

Using the hypotesis, we get:

$$\begin{aligned} I - 2\varepsilon &\leqslant M \leqslant I && (47)\\ 4\Sigma_\Delta(I - \varepsilon) &\leqslant \Sigma_{x|y} \leqslant 4\Sigma_\Delta && (48) \end{aligned}$$

from which

$$\begin{aligned} M &= I + O(\varepsilon)) && (49)\\ \Sigma_{x|y} &= 4\Sigma_\Delta(I + O(\varepsilon)) && (50) \end{aligned}$$

Then:

$$\begin{aligned} & \log|\Sigma_x| - \log|\Sigma_{x|y|} + x^t\Sigma_x^{-1}x - (x - My)^t\Sigma_{x|y}(x - My)\\ \approx\ & -(x - (I + O(\varepsilon))y)^t[4\Sigma_\Delta((I + O(\varepsilon))]^{-1}(x - (I + O(\varepsilon))y)\\ & - \log|4\Sigma_\Delta I + O(\varepsilon))| + \log|\Sigma_x| + x^t\Sigma_x x\\ & \to_{\varepsilon\to 0} -(x - y)4\Sigma_\Delta(x - y) - \log|4\Sigma_\Delta| \end{aligned}$$

where $x^t\Sigma_x^{-1}x$ is negligible with respect to the other terms for $\varepsilon \to 0$ (because $0 \leqslant x^t\Sigma_x^{-1}x \leqslant \varepsilon x^y\Sigma_\Delta^{-1}$) and $\log|\Sigma_x|$ is constant for every y

□

The matrix $\Sigma_\Delta = E_{p(x,y)}[x - (\frac{x+y}{2})]$ is the inner chunklet covariance matrix as defined in the Mahalnobis Distance, and then, under the conditions of the theorem 4.5.1 we obtain again the Coupled Method.

In the general case, otherwise, the formula that we get is different from the Mahalanobis Distance:

$$(x - My)^t \Sigma_{x|y}^{-1} (x - My) = (x - \mu_{x|y})^t \Sigma_{x|y}^{-1} (x - \mu_{x|y}) \tag{51}$$

The matrix M can be seen as a transformation that work on point y before applying the matrix $\Sigma_{x|y}^{-1}$.

In the case of the theorem, for example, M was close to the identity matrix (when the points in all pairs are very close to each other, the identity matrix is the best regression from one to the other), and the matrix distance was the Mahalanobis matrix.

To apply a transformation on data before using some distances for classifing or clustering, is a common procedure used in data analysis.

4.5.4 BLOSUM Score

In [32] the Blosum score (equation (3)) is analyzed from a mathematical perspective using the quantities of the Information Theory. We recall that in the formula (3) the f_{ij} corresponds to the observed relative frequency of amino acids inside the sequences of given proteins, while the probabilities p_{ij}, p_i and p_j need to be deduced from the population of proteins, and become a typical feature for this population.

Formula (3) is very similar to the formula of mutual information but here the frequencies f_{ij} are mixed with the *a priori* probabilities.

Rewriting the formula (3) to obtain the term $I(X, Y)$ (the mutual information between two protein X and Y), we gain factors relative to probabilities of the population. In particular, multiplying the argument of the logarithm by $\frac{f_{ij}}{f_{ij}} \frac{f_i f_j}{f_i f_j}$ the normalized score can be split:

$$S(X, Y) = I(X, Y) - D(F_{XY} \parallel P_{AB}) + D(F_X \parallel P_A) + D(F_Y \parallel P_B) \tag{52}$$

where $D(\cdot \parallel \cdot)$ is the Kullback Leibler Divergence, or Relative Entropy, F_{XY} is the joint frequency distribution of the amino acids pairs in the sequences (observed target frequencies), while F_X and F_Y respectively, are

the distributions of the amino acids inside X and Y (observed background frequencies). P_{AB} instead is the joint probability distribution associated to the Blocks database, and is the vector of target frequencies. Note also that $P_A = P_B = P$ is the probability distribution of the amino acids inside the same database Blocks (they are equal because $p(i,j) = p(j,i)$).

The set $\{I(X,Y), D(F_{XY} \parallel P_{AB}), D(F_X \parallel P_A), D(F_Y \parallel P_B)\}$ is named in [32], BLOSUM Spectrum (BLOSpectrum).

As already underlined in [32], this decomposition becomes important when we consider sequences for which the BLOSUM score indicates a weak or no correlation: when we use the BLAST program for comparing two sequences, indeed, the result of our research is the corresponding score value (plus additional information such as the E-value, etc). If the score value of the two sequences is too low, the program returns a phrase that say: "No significant similarity was found".

From the analysis of the BLOSpectrum, otherwise, we can understand the reasons of the low score: the high value of the term $D(F_{XY} \parallel P_{AB})$ togheter with low values for the $D(F_X \parallel P_A), D(F_Y \parallel P_B)\}$ terms, for example, indicates that the joint frequencies are not typical, i.e. are not close to the background distribution. This can be due also to a wrong choice of the model (the BLOSUMθ matrix) and the knowledge of the target frequency divergence may be of some help in identifying the best scoring matrix, that is the one tailored for the correct evolutionary distance.

If, on the other hand, changing the θ parameter does not affect $D(F_{XY} \parallel P_{AB})$, but the background amino acid frequencies ($D(F_X \parallel P_A)$ and $D(F_Y \parallel P_B)$) are always low, then the large target frequency divergence indicates some nonstandard evolutionary process (we are probably in the twilight zone: weakly correlated sequences with a very old common ancestor, or portions of proteins with strong structural properties that do not require the conservation of the entire sequence).

A more exhaustive analysis of different possible cases according to the different values of the BLOSpectrum is given in [32]

We only want to underline that this analysis compared with the previously described methods, shows the power of the alignment algorithms that not only use the information about the correlation between sequences (with the factor $I(X, Y)$) but also capture the evolutionary relations of the sequences using the divergence factor.

4.6 Results and Comparison

In this section we want to resume the results obtained using the main methods previously described.

We resume the results in the following tables; we construct a table for each of the datasets into which different methods have been applyed, while all the datasets processed by a single method are resumed in another single table.

The prediction quality of the different methods is usually examined by the resubstitution and jackknife methods. The use of these methods is on the one hand for testing the self consistency of the method, on the other hand for testing the results by cross-validation.

In general, the jackknife procedure (also called the leave one-out test) consists of removing a sequence from the training set, training the model with the remaining sequences and performing the test on the sequence removed. This process is tandemly repeated for all sequences in the training set, and the final prediction accuracy summarizes the outcome of all indipendent tests. Thus this procedure is considered the most appropriate for the assesment of a prediction method based on indipendent training and test data. Anyway, it is possible to use also a 10-fold cross-validation test that is statistically not so different from the jackknife procedure and less computational expensive [71].

If we want to test the self consistency of the distance, we must include in our method of attribution the reference sequence itself, otherwise, for testing the results, we must exclude that sequence from the calculation.

As can be seen from the tables, all the methods have overall rates of correct prediction higher than those of the simple geometry distance algorithms. To take into account the component coupled effect allowed the first real improvement in the class prediction problem.

Of course, the results by the resubstitution test are always higher than those by the jackknife procedure. Nonetheless, from the results in tables, it is possible to observe a decrease in the rate of correct predictions from the resubstitution to the jacknife test. This decrease is more remarkable for the component-coupled algorithm than for the simple geometry-distance algorithm, especially for small datasets (see for example table 4.8 and 4.9 for the two datasets of 138 and 253 proteins). This is because the component coupled algorithm needs more training data to make its prediction. Therefore, the information loss due to jackknifing will have a greater impact on the predicted results than those by the simple geometry-distance algorithm.

Nonetheless, sometimes low accuracy is obtained also in large dataset (see table 4.13). In [106] the authors have shown that the prediction of the four SCOP classes (all-α, all-β, α/β and $\alpha+\beta$) using Bayesian classification (that in section 4.5.2 we have seen to be equivalent to the Coupled Method) on non homologous sets of proteins is limited by 60%.

Although these results are considered controversial by some authors [116], it is possible to observe that low accuracy should be expected in non-homologous datasets. The higher results obtained with method different from the Coupled Method, sometimes are due to an incorrect procedure or implementation (see [71]).

According to the classification of protein structures in SCOP, all protein domains with more than 30% identity belong to the same protein family and must have the same structural class. Therefore, as already underlined by other researcheres (see [106] [71]) the structural class assignment of a new protein domain with homology $> 30\%$ to a protein of known structure can be also performed by sequence alignment. Any prediction method for the protein structural classes should only address those proteins for which low

homologous proteins are found in the Protein Data Bank.

A comparison with alignment algorithms is not done in these articles, except for [38] and [67]. In [38] two datasets of proteins are taken into account, as we have already described above: a dataset named **Chew-Kedem dataset** of 36 proteins belonging to five different families and another dataset of 86 non homologous protein families (SP-86 dataset).

The prediction quality is estimated by the ROC curves and corresponding AUC (Area Under the ROC curve) value or directly on a linearized matrix (see [38]) or on the output of some classifier algorithm (see [67]). In [38] in order to assess the performance of compression-based classification, via UPGMA and NJ under various compression algorithms, they have computed the F-measure against a gold standard.

The results show that the use of the USM methodology has the same performance and limitations as more standard methods such as global and local alignments and the linear correlation coefficient between the 20^k dimensional vectors of $k-$mer frequencies for $k = 1, 2, 3$. This seems to indicate that the main advantage of the USM methodology is its scalability with data size [38].

In particular, on the **SP-86 dataset**, every methods perform poorly on all classification tasks (class, architecture, topology, according to the CATH classification), a further example of low results obtained on non homologous dataset.

Low values of AUC are obtained also in [67] on the Dataset I, a selected dataset designed to evaluate distant sequence similarities.

In tables 4.5 and 4.6 we show some of the results in [38]. In [38] different kind of compressors are used, they can be divided into three group:

1. three state of the art tools for general purpose: Gzip, Bzip2 and PPMd algorithm

2. the memoryless compressors based on Huffman and Arithmetic Coding

3. the compressor based on the bwt transformation

We show only the higher F-measures achieved by each compressor belonging to the three group, in comparison with the alignment algorithms and the k-mer frequency correlation method.

In table 4.7 are shown the results of [22] on the **Nakashima_Dataset** of 131 proteins choosen from the original dataset of 135 proteins in [86] (the irregular folding type proteins have been left out because their number is only four, too small to have any statistical significance). No information on the homology level is given.

The percentages are refererred to the self consistency of the methods. The Mahalanobis Distance is compared with the usual geometric distances (Euclidean Distance, Minkowski's Distance) and the Discriminant Analysis. As we can see, the accuracy obtained with the Mahalanobis Distance is higher than that obtained by the other methods.

The only result regarding the jacknife procedure on this dataset known in literature is given in [106] where the Bayes decision rule equivalent to the Coupled Method, is used. The resubstitution procedure on this dataset is equal to 99.2%, confirming the better results with the use of the discriminant factor respect to the Mahalanobis Distance alone (94.7%), but the jacknife procedure gives a significantly lower prediction accuracy (42.7%). A plausible motivitation of this gap is given in [106] and is connected to the size and the low homology of the dataset. The authors in [106] use a dataset of 1189 proteins with identity level under 40% for constructing datasets of different sizes. On these non homologous datasets, they observe a rise in the accuracy of jackknife test with the size of the dataset and at the same time a decrese in the resubstitution value to a common value of 60% (see Fig 1 and table IV in [106]).

In table 4.8 and 4.9 we show the results obtained with the resubstitution and jacknife procedure respectively, on the datasets **138**, **253** and **510**. The Coupled Method is compared with the use of Hamming Distance and Euclidean Distance always on the aminoacidic composition vector. Only for the dataset 510 and 138 we have the comparison, even using the Coupled

Method, with the vector of autocorrelation functions [7].

The improvement in accuracy of jackknife test observed with the different datasets is explained in [21] by the dataset increase, while in [106] the cause is better attributed to the homologous proteins. Indeed, the 138 , 253 domains in these tables and the 359 domains in table 4.10 belong to 102, 129 and 130 protein families, respectively, i.e., in the dataset of 128 proteins, 36 are homologous, while in the other two, 124 and 229 respectively.

As already noted in [7] the use of the autocorrelation function method on small datasets gives a lower predictive accuracy (for both the resubstitution and jackknife tests) than that of the Coupled Method on the amino acid frequency vectors. This happens because the length of the attribute vector is equal to the appropriate number of autocorrelation functions, 30 in the article, whereas the number of the amino acid frequencies is only 20. They suggest an empirical minimal size of the training database of 200 proteins for applying their algorithm.

For the **Chou_dataset** of 359 proteins the results are obtained, other than by the geometric distances and the Coupled Method on the aminoacidic composition vectors, also with the Coupled Method on the autocorrelation functions of [7], with the use of discriminant analysis on the polypeptidi [78] and with the use of discrepancy measure [57] (results in table 4.10, 4.11).

The best accuracy is reached by the Discrepancy Measure [57], but we remember that these results are criticized in [64] for an improper implementation; with a right implementation the accuracy decreases both in the resubstitution and in jackknife test (see table 4.10, and 4.11).

The proteins come from the SCOP dataset and are highly homologous, and authors in [71] say that the high accuracy in this case is due to high homology (or improper procedure in the case of the discrepancy measure), while low accuracy shoul be expected for low homology sets, in agreement with [106].

Nonetheless, another dataset seems to controvert this claim. The accuracy values on the **204 non homologous-dataset** [18] are shown in table

4.12

This is a singular dataset on which other kind of methods are tested (such as SVM on pseudoaminoacid [15] or boosting algorithm [11]). We can see that the lower results are achieved by the Coupled Method on the amino acid frequencies (although under the hypotetical thereshold of 60% in [106]). The best results are obtained with the use of the Coupled Method on the pseudoaminoacid representation with aminoacid frequency and LZ value [110].

In general, all the good performing methods on this dataset, work on pseudoaminoacid rappresentations.

Low homology datasets are used in [57] and [78], but the results (shown in table 4.13) are probably due to improper procedure (see [71]). Finally, in table 4.13 we show the results for the remaining datasets described above.

The results on datasets PDB40-j and PDB40-j1 by the Discriminant Analysis on the polipeptidi composition [78] are obtained using the composition of the peptides selected from the discriminant analysis on the dataset PDB40-b.

4.6.1 Conclusions

From the results we can see that the use of the correlation matrix in the Mahalanobis Distance and also the incorporation of the eigenvalues in the coupled method allows a significant improvment respect to the traditional distances such as the euclidean or hamming distance on the aminoacid frequency vectors. These results show that the structural class is related to the amino acid composition of the corresponding sequences.

The other successfully applied representations are based on the polypeptides [78] [57] and auto-correlation functions computed for individual amino acid [7]. In particular, the results on homologous datasets, seem to point the ACF [7] as the best method (it takes into account the autocorrelation functions between amino acid index indicating various physicochemical and biochemical properties of amino acids). Nevertheless, using the autocorrelations with datasets containing low homology sequences resulted in signifi-

cantly worse results [71].

As already observed above, indeed, sequence homology is found to significantly affect prediction accuracy.

Promising results can be gained by the use of the coupled method with the pseudoaminoacid representation, in particular, we see as intersting the best result obtained on the **204 non homologous** dataset with the Coupled method on the pseudoaminoacid of 21 dimension given by the amino acid frequencies and the LZ complexity value of the protein.

The combination of the LZ complexity or compressor value of the proteins with other methods or codes seems to give promising results [67].

The comparison of the results is very hard either because different methods are tested on different datasets, and therefore no reliable and comprehensive comparison between different methods can be performed, or, when the datasets are the same, they are very small and characterized by unknown and most likely high sequence homology, which is shown to have significant impact on the prediction accuracy. Moreover, some algorithms use different procedures for measuring the quality of prediction (classification) of sequences into structural classes.

If we want to use these methods for extracting the significant features that allow a particular fold for a protein, the most intersting results are those on the non homologus dataset, for which also the alignment algorithms does not obtain good results, and for which it is more evident that the rules that regulate the fold of a protein are still unknown.

More promising methods on non homologous datasets in table 4.13 are those working on a significative peptides search. Though these results are criticized in [71], we think that this kind of code for the protein d (for clearly computational problems that this kind of coding involves for some methods, as, for istance, the coupled method).

We think that the use of a measure as the discrepancy [57], though a correct implementation of the method seems to cause succes fractions to decreas markably[64], has to the further investigated as well; from the split of

the BLAST score in the BLOSpectrum, we have seen that also the alignment algorihtms take into account the evolutionary divergence using the divergence between pair of amino acids in different sequences.

It would be very interesting to compare this methods with alignment algorithms and may be to develop other methods based on the extimation of the divergence of different sources, as the methods introduced in chapter 2.

In conclusion, over the last three decades many attempts were made to the prediction of protein structural classes task obtaining a progress in the resolution of the problem. At the same time, there was no comprehensive study that would point out some of the existing problems, which include testing standardization, introduction of new classification algorithms, and alternative sequence representations.

A good attempt in this direction has been made in [71], where it is shown a comprehensive, multigoal study that addresses comparison of different methods on datasets of homologous and non homologous proteins. In particular, eight different classification algorithms are used to perform structural class prediction: they include Bayesian classification, nearest neighbor, support vector machine, decision trees, rule based algorithms, neural networks, and logistic regression.

From the analysis made here, we think that the first intersting thing to do would be to compare the aforementioned methods, in particular those working on amino acid frequencies correlations (coupled method) or peptides frequencies or those taking into account of the amino acid order working on sequence amino acid correlations (compressors).

Both the methods should be compared with alignment algorithms based on BLOSUM score calculation.

BLOSUM score decomposition into its spectrum points out that alignment algortihms are constructed in such a way as to take into account correlations or divergence of sequences from a given evolution model based on known belonging structure sequences. The knowledge of this BLOSpectrum may be useful for the comparison of the alignment algorithms with the other

alignment-free methods, most of which, are based on similar quantities.

A full comparison between the aforementioned methods on a given non homologous dataset could be useful in finding a protein's features playing a bigger role in the protein fold.

The discovery of orthogonal methods could be useful in thinking of a methodologies combination.

CMP	Geometric Formula	Mean	Nearest Neighboor	Vertical Voting
		N=1	N=6-7	N=1
		82.35%	91.18%	85.29%
	Alpha=2 Formula	Mean	Nearest Neighboor	Vertical Voting
		N=1	N=5-13	N=1
		82.35%	88.24%	82.35%

Diagonal	Geometric Formula	Mean	Nearest Neighboor	Vertical Voting
		N=2-16	N=2-3	N=2
		88.24%	88.24%	88.24%
	Alpha=2 Formula	Mean	Nearest Neighboor	Vertical Voting
		N=4	N=2	N=2
		73.53%	88.24%	91.18%

Rows	Geometric Formula	Mean	Nearest Neighboor	Vertical Voting
		N=5	N=5-20	N=3, N=5-20
		88.24%	82.35%	82.35%
	Alpha=2 Formula	Mean	Nearest Neighboor	Vertical Voting
		N=9	N=10-19	N=2
		91.18%	91.18%	85.29%

[70]	USM formula	Mean	Nearest Neighboor	Vertical Voting
		66.7%	86.11%	83.33%

Table 4.2 ⋄ Comparison of the n-gram formula with the USM formula on the database Chew-Kedem [70].

Methods	F-measure
Gzip	0.8196
PPMd4/8/16	0.9605
Rc slow/med	0.8447
BwtRleHuff/BwtRleRc fast	0.8778
Geometric Formula N=4	0.8866
Alpha=2 Formula N=3	0.8788
LZ distance	0.80
salign-BLOSUM62-local	0.9849
salign-PAM120-global	0.9533

Table 4.3 ◇ F-measure on the CK-dataset. The trees are calculated via UPGMA cluster algorithm.

Methods	F-measure
Gzip	0.5450
PPMd2	0.5528
Rc slow	0.5461
BwtRleAc fast	0.5593
Geometric Formula N=3	0.5491
Geometric Formula N=6	0.5507
Alpha=2 Formula N=2	0.5425
Alpha=2 Formula N=6	0.5409
LZ distance	0.5433
salign-BLOSUM62-local	0.5391
salign-PAM120-global	0.5488

Table 4.4 ◇ F-measure on the SP-dataset. The trees are calculated via UPGMA cluster algorithm.

Ref.	Method	Dataset		F-measure
		Size	Homology	
Sequences				
[38]	NCD with PPMd8	36 (CK-36-PDB)	Unknown but homologous	0.9605%
	NCD with RC Slow	36 (CK-36-PDB)	Unknown but homologous	0.8447%
	NCD with BwtRleRc fast	36 (CK-36-PDB)	Unknown but homologous	0.8778%
Tops Strings of secondary structure elements				
[38]	NCD with PPMd8	36 (CK-36-PDB)	Unknown but homologous	0.9030%
	NCD with RC Slow	36 (CK-36-PDB)	Unknown but homologous	0.9030%
	NCD with BwtRleRc fast	36 (CK-36-PDB)	Unknown but homologous	0.8706%

Tops Strings with contact map				
[38]	NCD with PPMd8	36 (CK-36-PDB)	Unknown but homologous	0.9030%
	NCD with Ac Med	36 (CK-36-PDB)	Unknown but homologous	0.8881%
	NCD with BwtRleRc slow	36 (CK-36-PDB)	Unknown but homologous	0.8881%
[70]	USM on contact map	36 (CK-36-PDB)	Unknown but homologous	0.92%
Sequences				
	salign-BLOSUM62-local	36 (CK-36-PDB)	Unknown but homologous	0.9849%
	cor-word-2	36 (CK-36-PDB)	Unknown but homologous	0.5837%

Table 4.5 ⋄ Some F-measure via UPGMA of [38]. For the description of methods see [38]

Ref.	Method	Dataset		F-measure
		Size	Homology	
	Sequences			
[38]	NCD with PPMd2	86 (SK-86-PDB)	Unknown but homologous	0.5528%
	NCD with RC Slow	86 (SK-86-PDB)	Unknown but homologous	0.5461%
	NCD with BwtRleRc fast	86 (SK-86-PDB)	Unknown but homologous	0.5593%
	Atom Line from PDB file			
[38]	NCD with Bzip2	86 (SK-86-PDB)	Unknown but homologous	0.5779%
	NCD with RC Fast	86 (SK-86-PDB)	Unknown but homologous	0.5625%
	NCD with BwtMtfRleRc fast	86 (SK-86-PDB)	Unknown but homologous	0.5791%
	salign-PAM120-global	86 (SK-86-PDB)	Unknown but homologous	0.5488%
	cor-word-1	86 (SK-86-PDB)	Unknown but homologous	0.5447%

Table 4.6 ⋄ Some F-measure via UPGMA of [38]. For the description of methods see [38]

Ref.	Method	Dataset		Accuracy
		Size	Homology	
	Resubstitution			
[106]	Bayes decision rule (equivalent to Coupled Method)	131	Unknown	99.2%
[22]	Mahalanobis Distance	131	Unknown	94.7%
[23]	Minkowski's Distance	131	Unknown	79.7%
[66]	Discriminant Analysis	131	Unknown	76.5%
[86]	Euclidean Distance	135	Unknown	70.2%
	Jacknife			
[106]	Bayes decision rule (equivalent to Coupled Method)	131	Unknown	42.7%

Table 4.7 ⋄ Comparison of results on the **Nakashima_Dataset** (see previous section for the description)

Resubstitution Test

Ref.	Method	Dataset		Accuracy
		Size	Homology	
[21]	Coupled Method	138	Unknown	97.3%
	Euclidean Distance	138	Unknown	57.25%
	Hamming Distance	138	Unknown	55.80%
[7]	Coupled Method on the autocorrelation functions	138	Unkown	63.8%
[21]	Coupled Method	253	Unknown	95.26%
	Euclidean Distance	253	Unknown	53.36%
	Hamming Distance	253	Unknown	52.57%
[7]	Coupled Method on the autocorrelation functions	510	Unkown	98.2%

Table 4.8 ⋄ Results on three different datasets with the Resubstitution test. The reppresentations of the proteins are the AA composition vector (except that in [7]).

Jackknife Test

Ref.	Method	Dataset		Accuracy
		Size	Homology	
[21]	Coupled Method	138	Unknown	63.77%
	Euclidean Distance	138	Unknown	46.38%
	Hamming Distance	138	Unknown	48.55%
[7]	Coupled Method on the autocorrelation functions	138	Unkown	57.2%
[21]	Coupled Method	253	Unknown	79.05%
	Euclidean Distance	253	Unknown	50.20%
	Hamming Distance	253	Unknown	48.62%
	Mahalanobis Distance	253	Unknown	64.82%
[7]	Coupled Method on the autocorrelation functions	510	Unkown	91.8%
[21]	Coupled Method	510	Unknown	86.47%
	Euclidean Distance	510	Unknown	47.25%
	Hamming Distance	510	Unknown	47.06%

Table 4.9 ⋄ Results on three different datasets with the jacknife procedure. The reppresentations of the proteins are the AA composition vector (except that in [7] where the autocorrelation functions are used).

Resubstitution test

Ref.	Method	Dataset		Accuracy
		Size	Homology	
[21]	Coupled Method	359	Unknown but homologous	94.43%
	Euclidean Distance	359	Unknown but homologous	52.37%
	Hamming Distance	359	Unknown but homologous	55.15%
[7]	Coupled Method on the autocorrelation functions	359	Unknown but homologous	96.7%
[78]	Discriminant Analysis on the polipeptide composition	359	Unknown but homologous	99.7%
[57]	Discrepancy measure (l=3)	359	Unknown but homologous	100%
[64]	Criticated Discrepancy measure	359	Unknown but homologous	76.51%

Table 4.10 ⋄ Results of the Resubstitution test obtained with different methods on the homologous **Chou_Dataset** of 359 proteins. In the last row the results show in [64] respect to that obtained in [57]

Jackknife Test

Ref.	Method	Dataset		Accuracy
		Size	Homology	
[21]	Coupled Method	359	Unknown but homologous	84.12%
	Euclidean Distance	359	Unknown but homologous	41.22%
	Hamming Distance	359	Unknown but homologous	52.37%
[7]	Coupled Method on the autocorrelation functions	359	Unknown but homologous	90.5%
[57]	Discrepancy measure (l=3)	359	Unknown but homologous	95.8%
[64]	Criticated Discrepancy measure	359	Unknown but homologous	57.23%

Table 4.11 ⋄ Results of the jackknife procedure obtained with different methods on the homologous **Chou_Dataset** of 359 proteins. In the last row the results show in [64] respect to that obtained in [57]

Jackknife Test

Ref.	Method	Dataset		Accuracy
		Size	Homology	
[110]	Coupled Method on pseudoamin (AA compos. vector and LZ)	204	$< 30\%$	89.7%
[15]	SVM on pseudoamin (with AA compos. and corr. functions)	204	$< 30\%$	85.3%
[11]	boosting algorithm	204	$< 30\%$	83.8%
[110]	Mahalanobis Distance with correlation analysis approach	204	$< 30\%$	80.9%
[18]	Coupled Method	204	$< 30\%$	77%

Table 4.12 ⋄ Results of the jackknife procedure on the **204 non homologous dataset**

Jacknife Test				
Ref.	Method	Dataset		Accuracy
		Size	Homology	
[78]	Discriminant Analysis on polipeptide composition	758 (PDB40-j)	$< 40\%$	85.8%
	Discriminant Analysis on polipeptide composition	372 (PDB40-j1)	$< 40\%$	82.3%
[78]	Discriminant Analysis on polipeptide composition	1054 (PDB40-b)	$< 40\%$	75.2%
[21]	Coupled Method on AA composition	1054 (PDB40-b)	$< 40\%$	55.8%
[57]	Discrepancy measure (l=3)	1401 (T30-1401_Dataset)	$< 30\%$	75.02%
[64]	Discrepancy measure (critica)	1401 (T30-1401_Dataset)	$< 30\%$	63.88%
[57]	Discrepancy measure (l=3)	1530 (T10-1530_Dataset)	$< 10\%$	70.1%
[57]	Discrepancy measure (l=3)	1564 (T20-1564_Dataset)	$< 20\%$	72.4%
[57]	Discrepancy measure (l=3)	3998 (T90-3998_Dataset)	$< 90\%$	71.3%
Resubstitution Test				
[78]	Discriminant Analysis on polipeptide composition	1054 (PDB40-b)	$< 40\%$	91.7%
[21]	Coupled Method on AA composition	1054 (PDB40-b)	$< 40\%$	66.2%
[57]	Discrepancy measure (l=3)	1401 (T30-1401_Dataset)	$< 30\%$	99.46%
[64]	Discrepancy measure(critica)	1401 (T30-1401_Dataset)	$< 30\%$	95.86%

Table 4.13 $\diamond$ Results of the jackknife and resubstitution test on differents datasets

Bibliography

[1] S. F. Altschul et al "Basic Local Alignment Search tool", *J. Mol. Biol.*, **215**, 403-410 (1990)

[2] C. B. Anfinsen "Studiees on the principles that govern the folding of protein chains", Nobel Lecture, (1972)

[3] D. Benedetto, E. Caglioti, V. Loreto "Language Tree and Zipping", *Physical Review Letters*, **88**, no.4 (2002)

[4] C. H. Bennett, P. Gács, M. Li, P. M. B. Vitányi, W.H. Zurek "Information Distance", *IEEE Transaction on Information Theory*, **44**, no.4, 1407-1423 (1998)

[5] W. R. Bennett *Scientific and engineering problem-solving with the computer*, Prentice-Hall, Inc., Englewood Cliffs; New Jersey (1976)

[6] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, P. E. Bourne "The Protein Data Bank", ıNucleic Acids Research, **28**, 235-242 (2000)

[7] W.-S. Bu, Z.-P. Feng, Z. Zhang, C.-T. Zhang "Prediction of protein (domain) structural classes based on amino-acid index", *Eur.J.Biochem.*, **266**, 1043-1049 (1999)

[8] M. Burrows, D. J. Wheeler "A Block-sorting Lossless Data Compression Algorithm", *Technical report, DIGITAL System Research Center* (1994)

[9] H. Cai, S. R. Kulkarni, S. Verdu "Universal Divergence Estimation for Finite-Alphabet Sources", *IEEE Transaction of Information Theory*, **52**, n 8, 3456-3475 (2006)

[10] H. Cai, S. R. Kulkarni, S. Verdú "Universal entropy estimation via block sorting", *IEEE Trans. Inf. Theory.*, **50**, n 7, 1551-1561 (2004)

[11] Y.-D. Cai, K.-Y. Feng, W.-C. Lu, K.-C. Chou "Using LogistBoost classifier to predict protein structural class", *Journal of Theoretical Biology*, **238**, 172-176 (2006)

[12] A.J. Camm and L. Fei "Chronotropic incompetence–Part I: Normal regulation of the heart rate", *Clin Cardiol.* **19(5)**, 424-428 (1996)

[13] C. Cammarota, E. Rogora *Independence and symbolic independence of nonstationary heartbeat series during atrial fibrillation*, Physica A, **353**, 323-335 (2005)

[14] C. Cammarota, E. Rogora *Testing independence in time series via universal distributions of permutations and word*, Int. Jour. of Bif. and Chaos, **15**, 1-9 (2005)

[15] C. Chen, Y.-X. Tian, X.-Y. Zou, P.-X. Cai, J.-Y. Mo "Using pseudo-amino acid composition and support vector machine to predict protein structural class", *Journal of Theoretical Biology*, **243**, 444-448 (2006)

[16] X. Chen, S. Kwong, M. Li "A Compression Algorithm for DNA Sequences and its Applications in Genome Comparison", *RECOMB '00: Proceedings of the fourth annual international conference on Computational molecular biology*, (2000)

[17] C. Chothia, M. Levitt, D. Richardson "Structures of proteins: packing of $\alpha-$ helices and $\beta-$sheets", *Proc. Nat. Acad. Sci U.S.A*, **74**, 4130-4134 (1977)

[18] K.-C. Chou "A key driving force in determination of protein structural classes", *Biochem. Biophys. Research Commun.*, **264**, 216-224 (1999)

[19] K.-C. Chou " Prediction of Protein Structural Classes and Subcellular Locations", *Current Protein and Peptide Science*, **1**, n 2, 171-208 (2000)

[20] K.-C. Chou "Prediction of protein cellular attributes using pseudo-amino acid composition", *Proteins: Structure, Function and Genetics*, **43**, 246-255 (2001)

[21] K.-C. Chou, G. M.Maggiora "Domain structural class prediction", *Protein Engineering*, **11**, n 7, 523-538 (1998)

[22] K.-C. Chou, C.-T. Zhang "Predicting Protein Folding Types by Distance Functions That Make Allowances for Amino Acid Interactions", *The Journal of Biological Chemistry*, **269**, n 35, 22014-22020 (1994)

[23] P. Y. Chou " Prediction of Protein Structure and the Principles of Protein Conformation", *Plenum Press, New York*, 549-586 (1989)

[24] R. Cilibrisi, P. M. B. Vitányi "Clustering by compression", *IEEE Transaction on Information Theory*, **51**, no. 4, 1523-1545 (2005)

[25] T.M. Cover, J.A.Thomas, *Elements of Information Theory*, New Jork: Wiley (1991)

[26] M. O. Dayhoff, R. M. Schwartz, B. C. Orcutt "A model of evolutionary change in proteins", *Atlas of Protein Sequence and Structure*, **5**, supplement 3, 345-352, National Biomedical Research Foundation, Washington, DC, USA (1978)

[27] M. Degli Esposti, C. Farinelli, M. Manca and A. Tolomelli "A similarity measure for biological signals: new applications to HRV analysis", *JP J Biostat.*, **1**, n 1, 53-78 (2007)

[28] M. Degli Esposti, C. Farinelli and G. Menconi "Sequence distance via parsing complexity: Heartbeat signals", *to appear on Chaos Solitons & Fractals* (2007)

[29] M.H. DeGroot, *Probability and Statistics*, 2nd ed., Addison-Wesley Publishing Company, Reading MA 267-278 (1986)

[30] R. O. Duda and P. E. Hart *Pattern Classification and Scene Analysis*, a Wiley-Interscience Pubblication (1973)

[31] M. Effros, K. Visweswariah, S. R. Kulkarni, S. Verdú "Universal lossless source coding with the Burrows Wheeler Transform", *IEEE Transactions on Information Theory*, **48**, n 5, 1061-1081 (2002)

[32] F. Fabris, A. Sgarro, A. Tossi "Splitting the BLOSUM Score into Numbers of Biological Significance", *EURASIP Journal on Bioinformatics and System Biology*, (2007)

[33] M. Fahim "Cardiovascular sensory receptors and their regulatory mechanisms", *Indian J Physiol Pharmacol.*, **47**, 124-146 (2003)

[34] P. Fariselli, M. Compiani, R. Casadio "Predicting secondary structures of membrane proteins with neural networks", *Eur Biophys J*, **22**, 41-51 (1993)

[35] P. Fariselli, PL. Martelli, R. Casadio "A new decoding algorithm for hidden Markov models improves the prediction of the topology of all-beta membrane proteins", *BMC Bioinformatics*, **6**, Suppl 4:S12 (2005)

[36] S. Fazio, E.A. Palmieri, G. Lombardi G, B. Biondi "Effects of thyroid hormone on the cardiovascular system", *Recent Prog Horm Res.* **59**,31-50 (2004)

[37] Z.-P. Feng "An overview on predicting the subcellular location of protein", *In Silico Biology*, **2**, n 3, 291-303 (2002)

[38] P. Ferragina, R. Giancarlo, V. Greco, G. Manzini, G. Valiente "Compression-based classification of biological sequences and structus via the Universal Similarity Metric: experimental assessment", *BMC Bioinformatics*, **8**, n 252, 1471-2105 (2007)

[39] R. D. Finn, J. Mistry, B. Schuster-Böckler, S. Griffiths-Jones, V. Hollich, T. Lassmann, S. Moxon, M. Marshall, A. Khanna, R. Durbin, S.R. Eddy, E.L.L. Sonnhammer and A. Bateman "Pfam: clans, web tools and services", *Nucleic Acids Research*, **34**, D247-D251 (2006)

[40] R. Freeman "Assessment of cardiovascular autonomic function", *Clin Neurophysiol.* (2006)

[41] A. Godzik " The structural alignment between two proteins: Is there a unique answer?", *Protein Science*, **5**, 1325-1338, (1996)

[42] P. Grunwald, P. Vitanyi "Shannon Information and Kolmogorov complexity", *http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0410002* (2004)

[43] Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng CK, Stanley HE. "PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals", *Circulation*, **101**, n 23, e215-e220 [Circulation Electronic Pages; http://circ.ahajournals.org/cgi/content/full/101/23/e215] (2000)

[44] J. Gough, K. Karplus, R. Hughey, C. Chothia " Assignment of homology to genome sequences using a library of hidden Markov models that represent all proteins of known structure." , *J Mol Biol.*, **313**, n 4 903-19 (2001)

[45] P. Grassberger, A. Kraskov, W. Nadler, H. Stögbauer, "Comment on "Linguistic Analysis of th Human Heartbeat Using Frequency and Rank Order Statistics", *Physical Review Letters*, **92**, n 10 (2004)

[46] R. W. Hamming "Error Detecting and Error Correcting Codes", *Bell System Technical Journal*, **26**, n 2, 147-160 (1950)

[47] "Heart rate variability. Standards of measurement, physiological intepretation, and clinical use. Task force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology", *Circulation*, **93** 1043-1065 (1996)

[48] S. Henikoff, J. G. Henikoff "Amino acid substitution matrices from protein blocks", *Proc. Natl. Acad. Sci. USA*, **89**, 10915-10919 (1992)

[49] A. B. Hillel, D. Weinshall "Learning Distance Function by Coding Similarity, *Proceedings of the* 24^th *International Conference on Machine Learning ICML '07: Proceedings of the 24th international conference on Machine learning*, **227**, 65-72 (2007)

[50] http://www.ebi.ac.uk/swissprot/index.html

[51] http://www.physionet.org/physiobank/database/chf2db/

[52] http://www.physionet.org/physiobank/database/nsrdb/

[53] http://www.physionet.org/physiobank/database/nsr2db/

[54] J. Hu, X. Shen, Yu Shao, C. Bystroff, M. Zaki,"Mining Protein Contact Maps", *in Proc. 2nd BIOKDD Workshop Data Mining Bioinformatics* (2002)

[55] P.Ch. Ivanov, M.G. Rosenblum, C.-K. Peng, J.E. Mietus, S. Havlin, H.E. Stanley and A.L. Goldberger "Scaling and universality in heart rate varibability distributions ", *Physica A*, **249**, 587 (1998)

[56] J.R. Jennings JR and M.W. van der Molen "Cardiac timing and the central regulation of action", *Psychol Res.* **66(4)**, 337-349 (2002)

[57] L. Jin, W. Fang, H. Tang "Prediction of protein structural classes by a new measures of information discrepancy", *Computational Biology and Chemistry*, **27**, 373-380, (2003)

[58] R. A. Johnson, D. W. Wichern *Applied Multivariate Statistical Analysis*, Prentice-Hall, Inc. Eglewood Cliffs, N.J. (1982)

[59] J. Jung, B. Lee "Protein structure alignment using enviromental profiles", *Protein Eng*, **13**, pp 535-543, (2000)

[60] M. Kac "On the notion of recurrence in discrete stochastic processes", *Bull. of the Amer. Math. Soc.*, **39**, 78-83 (1947)

[61] J. Kalda, M. Sakki, M. Vainu, M. Laan. "Non-linear and scale-invariant analysis of the Heart Rate Variability", Arxiv.org physics/0303041 (2003)

[62] A. Kaltchenko "Algorithms for estimating information distance with aapplication to bioinformatics and linguistics", *Proceedings of the 2004 Canadian Conference on Electrical and Computer Engineering*, **4**, 2255-2258 (2004)

[63] J.-W. Kantelhardt, Y. Ashkenazy, P.-Ch. Ivanov, A. Bunde, S. Havlin, T. Penzel, J.-H. Peter, H.-E. Stanley " Characterization of sleep stages by correaltions in the magnitude and sign of heartbeat increments", *Physical Revie E* **65** 1-6 (2002)

[64] K. D. Kedarisetti, L. Kurgan, S. Dick "A comment on "Prediction of protein structural classes by a new measures of information discrepancy"", *Computational Biology and Chemistry*,**30**, 393-394 (2006)

[65] V. Keselj, F. Peng, N. Cercone, C. Thomas "N-gram-based author profiles for authorship attribution", *Proceedings of the Conference Pacific Association for Computational Linguistics, PACLING'03*, 255-264 (2003)

[66] P. Klein "Prediction of protein structural class by discriminant analysis", *Biochim. Biophys. Acta*, **874**, 205-215 (1986)

[67] A. Kocsor, A. K. Farkas, L. Kajan and S. Pongor "Application of compression-based distance measures to protein sequence classification: a methodological study", *Bioinformatics*, **22**, n 4, 407.412 (2006)

[68] S. R. Kosaraju, G. Manzini "Compression of low entropy strings with Lempel Ziv algorithms", *SIAM J. Comput.*, **29**, n 3, 893-911, (1999)

[69] P. Koehl "Protein Structure Similarities", *Structural Biology*, **11**, 348-353, (2001)

[70] N. Krasnogor, D. A. Pelta "Measuring the similarity of protein structures by means of the universal similarity metric", *Bioinformatics*, **20**, n 7, 1015-1021 (2004)

[71] L. A. Kurgan, L. Homaeian "Prediction of structural classes for protein sequences and domains-Impact of prediction algorithms, sequence representation and homology, and test procedures on accuracy", *Pattern Recognition*, **39**, 2323-2343 (2006)

[72] A. Lempel and J. Ziv " On the complexity of finite sequences", *IEEE Trans. Inform. Theory*, **IT-22**, n 1, 75-81 (1976)

[73] A. M. Lesk, C. Chothia, "How different amino acid sequences determine similar protein structures", *J Mol Biol*, **136**, 225-270, (1980)

[74] A. M. Lesk, C. Chothia, "The relation between the divergence of sequence and structure in proteins", *The EMBO Journal*, **5**, 823-826, (1986)

[75] M. Li, J. H. Badger, X. Chen, S. Kwong, P. Kearney and H. Zhang, "An information-based sequence distance and its application to whole mitochondrial genome phylogeny", *Bioinformatics*, **17**, no.2, 149-154 (2001)

[76] M. Li, X. Chen, X. Li, B. Ma, P. M. B. Vitányi "The Similarity Metric", *IEEE Transaction on Information Theory*, **Xx**, n Y, 1-13 (2004)

[77] M. Li and P. M. B. Vitányi, *An Introduction to Kolmogorov Complexity and its Applications*, Springer-Verlag, New York, 2nd Edition (1997)

[78] R.-Y. Luo, Z.-P. Feng, J.-K. Liu "Prediciton of protein structural class by amino acid and polypeptide composition", *Eur.J.Biochem.*, **269**, 4219-4225 (2002)

[79] G. Manzini "The Burrows-Wheeler Transform: Theory and Pratice", *Lecture Notes in Computer Science*, **1672**, 34-47, Springer, Berlin (1999)

[80] D. Makowiec, R. Galska, A. Dudkowska, A. Rynkiewicz and M. Zwierz "Long-range dependencies in heart rate signals-revisited", *Physica A*, **369**, 632-644 (2006)

[81] N. Marwan, N. Wessel, U. Meyerfeldt, A. Schirdewan, J. Kurths. " Recurrence Plot Based Measures of Complexity and its Application to Heart Rate Variability Data", *Physical Review E*, **66**, 026702-1: 026702-8, (2002)

[82] M. Meyer and O. Stiedl "Self-affine fractal variability of human heartbeat interval dynamics in health and disease ", Eur. J. Appl. Physiol., **90**, 305-316 (2003)

[83] CE Metz "Basic principles of ROC analysis", *Seminars in Nuclear Medicine*, **8**, 283-298 (1978)

[84] J.E.Mietus , C.K. Peng , I. Henry I, R. L. Goldsmith and A. L. Goldberger "The pNNx files: re-examining a widely used heart rate variability measure. ", *Heart*, **88**, n 4, 378-380 (2002)

[85] A. G. Murzin, S. E. Brenner, T. Hubbard, C. Chothia "SCOP: a structural classification of protein database for the investigation of sequences and structures", *J. Mol. Biol.*, **247**, 536-540 (1995)

[86] H. Nakashima, K. Nishikawa, T. Ooi "The folding type of a protein is relevant to the amino acid composition" ,*J Biochem*, **99**, n1, 153-62 (1986)

[87] C.A. Orengo "COR-Topological fingerprints for proteins structural families", *Protein Sci*, **8**, 699-715 (1999)

[88] D. Ornstein, B. Weiss "Entropy and data compression", *IEEE Trans. Inform. Th*, **IT-39**, 78-83 (1993)

[89] K. Otsuka, S. Murakami , Y. Kubo, T. Yamanaka, G. Mitsutake, S. Ohkawa , K. Matsubayashi, S. Yano, G. Cornelissen and F. Halberg " Chronomics for chronoastrobiology with immediate spin-offs for life quality and longevity", *Biomed Pharmacother.*, **57** 1-18 (2003)

[90] H. Otu, K. Sayood "A new sequence distance measure for phylogenetic tree construction", *Bioinformatics*, **19**, n 16, (2003)

[91] F. Pearl et al. "The CATH Domain Structure Database and Related Resources Gene3D and DHS Provide Comprehensive Domain Family Information for Genome Analysis", *Nucleid Acids Res*, **33**, n D, D247-D251 (2005)

[92] W. R. Pearson, M. L. Sierk "The limits of protein sequence comparison?", *Structural Biology*, **15**, 254-260 (2005)

[93] C.K. Peng, S. Havlin, H.E. Stanley and A.L. Goldberger "Quantification of scaling exponents and crossover phenomena in nonstationary heartbeat time series", *Chaos*, **5**, 82 (1995)

[94] A. Puglisi, D. Benedetto, E. Caglioti, V. Loreto, A. Vulpiani "Data compression and learning in time sequences analysis", *Physica D*, **180**, 92-107 (2003)

[95] A.K. Reyners, B.P. Hazenberg, W.D. Reitsma and A.J. Smit " Heart rate variability as a predictor of mortality in patients with AA and AL amyloidosis", *Eur Heart J.*, **23(2)** 157-161 (2002)

[96] J. Rocha, F. Rossello, J. Segura "Compression ratios based on the Universal Similarity Metric still yield protein distances far from CATH distances", http://www.citebase.org/abstract?id=oai:arXiv.org:q-bio/0603007 (2006)

[97] P. C. Shields, *The ergodic theory of discrete sample paths*, Graduate Studies in Mathematics, vol. 13, AMS (1996)

[98] M. L. Sierk, W. R. Pearson "Sensitivity and selectivity in protein structure comparison", *Protein Science*, **13**, 773-785 (2004)

[99] A. P. Singh, D. L. Brutlag "Protein Structure Alignment: A Comparison of Methods", http://cmgm.stanford.edu/brutlag/Abstracts/...singh00.html (2000)

[100] T. Sing, O. Sander et al., "ROCR:visualizing classifier performance in R", *Bioinformatics*, **21** n 20, 3940-3941 (2005)

[101] T. F. Smith, M. S. Waterman "Identification of common molecular subsequences", *J. Mol. Biol.*, **147**, 195-197 (1981)

[102] M. Suyama, Y. Matsu, K. Nishikawa "Comparison of protein structures using 3D.profile alignment", *J Mol Evol*, **44**, S163-S173, (1997)

[103] M. C. Teich, S. B. Lowen, B. M. Jost, K. Vibe-Rheymer and C. Heneghan "Heart Rate Variability: Measures and Models", *Nonlinear Biomedical Signal Processing*, **II**, Dynamic Analysis and Modeling; edited by M. Akay, 159-213 (2001)

[104] The R Development Core Team, "R: A Language and Enviroment for Statistical Computing-Reference Index", Version 2.6.1, 26/11/2007

[105] A. Tomović, P. Janicić, V. Keselj "N-gram based classification and Hierarchical Clustering of Genome Sequences", *Computer Methods and Programs in Biomedicine*, **81**, n 2, 137-153 (2006)

[106] Z.-X. Wang, Z. Yuan "How good is prediction of protein structural class by the component coupled method?", *PROTEINS: Structure, Function, and Genetics*, **38**, 165-175 (2000)

[107] M. Wölfel, H. K. Ekenel "Feature Weighted Mahalanobis Distance: Improved Robustness for Gaussian Classifiers", *13th European Signal Processing Conference, EUSIPCO 2005*, (2005)

[108] D. G. Wyse "Rate control vs rhythm control strategies in atrial fibrillation", *Prog Cardiovasc Dis.* **48**, n 2, 125-138 (2005)

[109] X. Xiao, S. Shao, Y. Ding, Z. Huang, Y. Huang, K.-C. Chou "Using complexity measure factor to predict protein subcellular location", *Amino Acids*, **28**, 57-61 (2005)

[110] X. Xiao, S.-H. Shao, Z.-D. Huang, K.-C. Chou "Using Pseudo Amino Acid Composition to Predict Protein Structural Classes: Approached with Complexity Measure Factor", *Journal of Computational Chemistry*, **27**, n 4, 478-482 (2006)

[111] A. Yang, S.-S. Hseu, H.-W. Yen, A. L. Goldberger and C.-K.Peng "Linguistic Analysis of the Human Heartbeat Using Frequency and Rank Order Statistics", *Physical Review Letters*, **90**, n 10, 1-4 (2003)

[112] M. E. Young "The circadian clock within the heart: potential influence on myocardial gene expression, metabolism, and function", *Am J Physiol Heart Circ Physiol.*, **290**, 1-16 (2006)

[113] J. Ziv, A. Lempel "Compression of individual sequences via variable-rate coding", *IEEE Trans. Inform. Theory*, **IT-24**, n 5, 530-536, (1978)

[114] J. Ziv, A. Lempel " A Universal Algorithm for Sequential Data Compression", *IEEE Transactions on Information Theory*, **23**, n 3, 337-343 (1977)

[115] J. Ziv , N. Merhav "A measure of relative entropy between individual sequences with application to universal classification", *IEEE Transaction of Information Theory*, **39**, n 4, 1270-1279 (1993)

[116] G. P. Zhou, N. A. Munt "Some Insights into Protein Structural Class Prediction", *PROTEINS: Structure, Function, and Genetics*, **44**, 57-59 (2001)