ANALYSIS OF THE SUBSEQUENCE COMPOSITION OF BIOSEQUENCES

A Thesis
Presented to
The Academic Faculty

by

Fabio Cunial

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in
Computational Science and Engineering

School of Computational Science and Engineering
Georgia Institute of Technology
August 2012

ANALYSIS OF THE SUBSEQUENCE COMPOSITION OF BIOSEQUENCES

Approved by:

Professor Alberto Apostolico, Advisor
School of Computational Science and Engineering
Georgia Institute of Technology

Professor Alexander Gray
School of Computational Science and Engineering
Georgia Institute of Technology

Professor Steve Harvey
School of Biology
Georgia Institute of Technology

Professor Christine Heitsch
School of Mathematics
Georgia Institute of Technology

Professor Sorin Istrail
Department of Computer Science
Brown University

Date Approved: May 2, 2012

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Measuring the amount of information and of shared information in biological strings, as well as relating information to structure, function and evolution, are fundamental computational problems in the post-genomic era. Classical analyses of the information content of biosequences are grounded in Shannon's statistical telecommunication theory, while the recent focus is on suitable specializations of the notions introduced by Kolmogorov, Chaitin and Solomonoff, based on data compression and compositional redundancy. Symmetrically, classical estimates of mutual information based on string editing are currently being supplanted by compositional methods hinged on the distribution of controlled substructures.

Current compositional analyses and comparisons of biological strings are almost exclusively limited to short sequences of contiguous solid characters. Comparatively little is known about longer and sparser components, both from the point of view of their effectiveness in measuring information and in separating biological strings from random strings, and from the point of view of their ability to classify and to reconstruct phylogenies. Yet, sparse structures are suspected to grasp long-range correlations and, at short range, they are known to encode signatures and motifs that characterize molecular families.

In this thesis, we introduce and study compositional measures based on the repertoire of distinct *subsequences* of any length, but constrained to occur with a predefined maximum gap between consecutive symbols. Such measures highlight previously unknown laws that relate subsequence abundance to string length and to the allowed gap, across a range of structurally and functionally diverse polypeptides. Measures

on subsequences are capable of separating only few amino acid strings from their random permutations, but they reveal that random permutations themselves amass along previously undetected, linear loci. This is perhaps the first time in which the vocabulary of all distinct subsequences of a set of structurally and functionally diverse polypeptides is systematically counted and analyzed.

Another objective of this thesis is measuring the quality of phylogenies based on the composition of sparse structures. Specifically, we use a set of repetitive gapped patterns, called *motifs*, whose length and sparsity have never been considered before. We find that *extremely sparse* motifs in mitochondrial proteomes support phylogenies of comparable quality to state-of-the-art string-based algorithms. Moving from *maximal* motifs – motifs that cannot be made more specific without losing support – to a set of generators with decreasing size and redundancy, generally degrades classification, suggesting that redundancy itself is a key factor for the efficient reconstruction of phylogenies. This is perhaps the first time in which the composition of all motifs of a proteome is systematically used in phylogeny reconstruction on a large scale.

Extracting all maximal motifs, or even their compact generators, is infeasible for entire genomes. In the last part of this thesis, we study the robustness of measures of similarity built around the dictionary of LZW – the variant of the LZ78 compression algorithm proposed by Welch – and of some of its recently introduced gapped variants. These algorithms use a very small vocabulary, they perform linearly in the input strings, and they can be made even faster than LZ77 in practice. We find that dissimilarity measures based on maximal strings in the dictionary of LZW support phylogenies that are comparable to state-of-the-art methods on test proteomes. Introducing a controlled proportion of gaps does not degrade classification, and allows to discard up to 20% of each input proteome during comparison.

CHAPTER I

THE SUBSEQUENCE COMPOSITION OF POLYPEPTIDES

Many approaches have been developed and tested over the years in an attempt at capturing the structure embodied in artifacts and natural objects alike. Despite these efforts, we still lack measures and meters to define and appraise this elusive attribute. The recent 50th anniversary issue of the Journal of the ACM opens, for example, with an essay by Frederick P. Brooks Jr. entitled "Three great challenges for half century old computer science". The author gives a list of outstanding problems, the first of which is described as follows [27]:

> Shannon and Weaver performed and inestimable service by giving us a definition of information and a metric for information as communicated from place to place. [...] We have no theory however that gives us a metric for the information embodied in structure. [...] I consider this missing metric to be the most fundamental gap in the theoretical underpinning of information and computer science.

Not surprisingly, Brooks points to biological sequences as the ideal test-bed for this endeavor: in biology, the transition to the molecular level has made centerpiece of the principle that form, interpreted as purely syntactic organization, dictates function. Moreover, the quantitative underpinning of the information content of biosequences is an obvious prerequisite to the quantitative modeling and study of biological function and evolution. Several past studies have addressed the question of what distinguishes biosequences from random strings: such studies typically analyze the organization of biosequences in terms of their constituent characters or subwords (i.e. blocks of consecutive text characters), and have consistently exposed a tenacious lack of

compressibility on behalf of biosequences. This chapter describes an assessment of the structure and randomness of polypeptides in terms on newly introduced parameters that relate to the vocabulary of their (suitably constrained) *subsequences* rather than their *substrings*. Our measures are seen to grasp structural/functional information in a dataset of biochemically diverse polypeptides, where they are related to each other under a specific set of rules. Measures on subsequences are capable of separating only few amino acid strings from their random permutations, but they show that *the random permutations of most polypeptides amass along specific linear loci*. This is perhaps the first time in which the vocabulary of all distinct subsequences of a set of structurally and functionally diverse polypeptides is systematically counted and analyzed.

1.1 Introduction

Defining and measuring the amount of information contained in biological strings, and relating this information to structure, function and chemical activity [32, 95], has long been an elusive problem, both for the inherent difficulty of formalizing intuitive notions of "information" [88] and for the peculiar structure of these strings.

Proteins, for example, are optimized by selection to assume specific chemical properties and spatial conformations: this streamlining tends to remove redundancies [4], yielding strings of amino acids in which every symbol carries information; such "slightly edited random strings" [174] are therefore hardly distinguishable from their random permutations when measured with both statistical and algorithmic definitions of information. Not even translating amino acids with scales that capture relevant physico-chemical properties provides significantly more insight: for example, the distribution of hydrophobicity – a key property influencing folding and spatial stability – along the sequence of most proteins is well known to be indistinguishable from random [152, 177]. The very presence of repetitions and redundancies has been implicated in

human diseases at the DNA level [24], and in the formation of toxic fibrillar structures at the protein level [28]. Repetitions in polypeptides have also been conjectured to multiply the folding possibilities by introducing many interactions with similar energy [171]: these alternatives would prevent the convergence of the folding process into a global minimum. Wet-lab experiments with random polypeptides [45, 131] have shown that secondary and tertiary structures do appear frequently and spontaneously in random strings built upon suitably small alphabets. Many of the basic folding patterns of natural proteins can even be explained theoretically by assuming the randomness of their primary sequence [176]: this seems to suggest that the main carrier of folding information is the *composition* of amino acids rather than their linear ordering [134]. All these clues, that nicely fit into the neutral theory of evolution, have oriented biochemists towards seeing modern proteins as memorized ancestral random polypeptides, that have been slightly edited by selection to optimize their active sites and stability under specific physiological conditions[1]. As Jaques Monod has put it [110]:

> In 1952, F. Sanger described the first complete sequence of a globular protein. This turned out to be both a revelation and a deception. This sequence, defining the structure and therefore the elective properties of a functional protein (insulin), did not show any regularity, characteristic feature, or limit. In those days it was hoped that, with the addition of further documentation, it might be possible to find the general laws of association and some functional correlations. Today we know hundreds of sequences corresponding to the proteins extracted from many different organisms. From them and their systematic comparison, performed with the help of up-to-date analysis and calculation devices, we can now deduce the general law: the chance law. More precisely, these structures are "random" because by knowing precisely the order of 199

[1] An exception to this universal rule of disorder is represented by strongly nonrandom polypeptides: about 25% of all amino acids in current databases are estimated to be in "low complexity", highly regular regions, and about 34% of all proteins in current databases are estimated to contain at least one such low complexity region [179]. These segments are routinely searched for and masked before local alignment searches [79, 178].

> residues of a protein containing two hundred it is not possible to formulate a theoretical or empirical law which allows us to predict the nature of the only residue still to be analytically identified.

Along with this intrinsic, evolutionary randomness, two additional problems make the definition of information in polypeptides even more elusive. The first problem is *context*: the translation of an amino acid string into a three-dimensional structure is made possible by the cooperation of many distant substrings of the same and of different molecules (e.g. chaperones, multimers); the transport of many proteins to their proper cellular compartment and the acquisition of their final function depend on multiple post-translational modifications. Therefore, the information that leads a protein to assume its specific biological role is distributed in a context of interactions that transcends the single sequence [1, 61, 88]. The second problem is *resolution*: the key functions of a protein are often implemented by few atoms configured in specific spatial arrangements and bearing specific chemical properties. A single letter of the primary sequence of a protein hides tens of such atoms, positions and properties: these sub-symbolic signals are doomed to evade any measure of information that treats proteins as strings on the traditional amino acid alphabet.

Notwithstanding these fundamental issues, the question of what and how much information is carried by amino acid sequences has historically attracted a lot of attention, both for obvious purposes of classification, prediction and insight into folding and evolution, and for the screening and synthesis of artificial polypeptides for their use in new drugs [44, 47]. Some successes have been recorded, especially in the context of large sets of non-homologous proteins (e.g. the proteome of an organism [2, 23, 107]). Estimates of differential entropy and context-free grammar complexity [174] have shown that the complexity of such large sets is lower than the complexity of a corresponding set of random strings by approximately 1%, about one third of which is caused by well-known low-complexity regions. Evidence of weak correlations

at short, medium, and long range has also been found: positive correlations appear at medium range (≥ 100) and decrease with distance, implying that the amino acid distribution of proteins that are close in the genome is more similar than that of proteins far in the genome. The sign of the correlation between pairs of amino acids at medium distance forms groups that resemble those traced by widely accepted physicochemical properties. Family-dependent, short-range periodicities in hydrophobicity, α-helix propensity and charge have also been detected [173], and have been attributed to interactions between elements of the same secondary structures.

Both in statistical and in algorithmic information theory, the search for correlations and patterns is intimately related to the construction of compact models [87, 90, 93]. Since a provocative 1999 study that advocated the incompressibility of proteomes [114], there has been a modest flourishing of compression techniques tuned for long concatenations of polypeptides, spanning both the substitutional and the statistical realms [64]. We mention, among others, techniques consisting in instantiating the PPM algorithm with contexts of multiple lengths, weighted by amino acid mutation probabilities [114]; searching for exact and approximate reverse complements, repeats, and weighted context trees [108]; partitioning amino acids according to their frequency and invoking popular text compressors [150]; using amino acid substitution matrices to guide the creation of Huffman codes [75]; building an offline dictionary of motifs with flexible gaps, constrained to be maximal in density and extension and to occur sufficiently frequently in the dataset [12]; using panels of weighted experts that estimate the probability of a symbol using Markov models encoding species information, local context information, as well as repeated and complementary reversed substrings [31]. These methods achieve entropies that range from about 3.67 to 4.05 bits per symbol, while other estimations based on the k-th order Shannon formula and Zipf curves reach 2.5 bps; incorporating secondary structure information in a gambling algorithm à la Shannon lowers this bound to about 2 bps [162].

As expected, the analysis of stand-alone sequences has yielded more limited results. Measures of entropy over sliding windows have been shown to separate globular and fibrous proteins [145], and Lempel-Ziv complexity has been used to predict the cellular location of proteins [182]. Adding physico-chemical information to amino acids has enabled a Fourier analysis to detect characteristic periodicities in two protein families with similar structural architectures [133]; a mapping of recoded protein sequences onto one-dimensional Brownian bridges has revealed systematic deviations from randomness that have been related to energy minimization [120]. The entropy of the primary sequence has also been shown to correlate with the inverse packing density and the hydrophobicity of residues in their spatial conformations [97].

In the present chapter, rather than identifying the information content of biosequences with their negentropy or compressibility, we follow a *compositional* approach that was probably proposed for the first time in [39], and which is likely to gain in importance with the current popularization of alignment-free algorithms for sequence comparison [169, 170]. Colosimo and De Luca [39] define the complexity of a biological string s as a function $f_s(n)$ that maps every integer $1 \leq n \leq |s|$ to the number of distinct substrings of length n that occur in s. In their experiments, they compare the DNA sequence s corresponding to a gene to its random reshuffle s': when all $O(|s|^2)$ substrings are taken into account, no significant difference between $f_s(n)$ and $f_{s'}(n)$ is reported. However, when only the $O(|s|)$ set of *right-maximal* substrings is considered (i.e. substrings that cannot be extended to the right with any symbol of the alphabet without losing at least one of their occurrences in s), a systematic difference between $f_s(n)$ and $f_{s'}(n)$ appears. Similar k-mer spectra have been studied more recently at a genomic scale: Chor et al. [35] consider function $f_s^k(n)$ that maps an integer $0 \leq n \leq |s| - k + 1$ to the number of distinct substrings of length k that occur n times in s, showing a multimodal behavior in selected genomes and reproducing it with first- and second-order Markov chains.

In this chapter we similarly relate information content to laws that govern the abundance of suitable combinatorial substructures of polypeptide strings. Rather than focusing on windows of fixed length or on substrings, however, we measure the composition of *subsequences* of any length. In contrast to the case of substrings, one difficulty when dealing with subsequences is that their number escalates quite rapidly and just as rapidly saturates the space of possible conformations, thereby turning the whole quest into a vacuous endeavor: we must look thus for words that appear as a *constrained* subsequence of the subject sequence. Our constraint consists of bounding the hiatus or interval ω between the text positions that may elapse between any two consecutive characters in one of our subsequences.

This chapter is organized as follows. Section 1.2 formalizes the notion of constrained subsequence and of class of positional equivalence. In the spirit of [39], Section 1.3 characterizes a set of subsequences as extremal, in the sense that they cannot be enriched with characters without losing some occurrence in the string. Subsequences and equivalence classes are then embedded in a natural spatial representation, in which they assume the form of paths and points, respectively. Even for small values of ω, the number of ω-subsequences can be exponential in the length of the host string: Section 1.4 describes an implicit representation taking quadratic space for finite alphabets. Sections 1.5 and 1.6 introduce suitable measures on this representation, and test such measures, respectively, on a small collection of artificial strings with various degree of structure, and on a set of polypeptides. Section 1.7 describes a previously unknown array of laws that, in the dataset of polypeptides, are seen to relate our measures to string length and to the hiatus of subsequences. Finally, Section 1.8 studies regularities that constrain pairs of measures in random permutations of our dataset.

1.2 Definitions, notation and state of the art

Given a nonempty string s from alphabet Σ, a *subsequence* of s is any string v that can be obtained by removing from s one or more, not necessarily consecutive characters. An *occurrence* of v in s is specified by a list of positions of s matching the characters of v consecutively. The positions of s that correspond to the first (respectively, last) character of v form the *left* (respectively, *right*) *occurrence list* of v, denoted by $\mathcal{L}_v = \{l_0, l_1, \ldots, l_{k-1}\}$ (respectively, $\mathcal{R}_v = \{r_0, r_1, \ldots, l_{h-1}\}$). Among all the occurrences of v as a subsequence of s, one is led to naturally privilege the leftmost.

Definition 1 ([52])**.** *Let I be the set of all occurrences of a string v as a subsequence of a string s. The* canonical *(or* leftmost*) occurrence of v in s is $\mathbf{i}^* = \langle i_0^*, i_1^*, \ldots, i_{|v|-1}^* \rangle \in I$ such that $i_j^* \leq i_j \; \forall \; \mathbf{i} = \langle i_0, i_1, \ldots, i_{|v|-1} \rangle \in I$, $0 \leq j < |v|$.*

The function that assigns to a nonempty string v its canonical occurrence in s is clearly bijective. The canonical occurrence, as well as the number of occurrences of all prefixes of v as a subsequence of s, can be computed in overall $O(|v| \cdot |s|^2)$ space and $O(|v| \cdot |s|^3)$ time using recurrence relations and dynamic programming [52] – ubiquitous tools in counts of subsequences.

A natural way to measure the complexity of s – reminiscent of the substring complexity studied in [35, 39] – is counting the number of its distinct subsequences (also known as *turbulence* in the social sciences [53]). The number of distinct subsequences of s can be computed in $O(|s|^2)$ time and space [52], and similar techniques allow to count the number of distinct subsequences of each length in a string, the number of distinct subsequences common to two strings, and the number of distinct subsequences of length at least k common to two strings, with comparable time and space complexities [52]. The maximum number of distinct subsequences in a string of length n on alphabet Σ satisfies the Fibonacci-like recurrence $M(n) = \sum_{k=1}^{|\Sigma|} M(n-k) + 1$, with $M(n) = 2^n \; \forall \; 0 \leq n \leq |\Sigma|$ [52]. More specifically, the string that maximizes the

number of distinct subsequences of every length among all strings of length n on Σ, is the length-n prefix of the infinite string σ^+, where $\sigma = \sigma_0\sigma_1 \ldots, \sigma_{|\Sigma|-1}$ is the string in which all characters of alphabet Σ occur exactly once in their total order [33, 59, 76]. Conversely, $\sum_{i=0}^{n-k} \binom{\tau-n+k}{i}$ is a tight *lower bound* on the number of distinct subsequences of length k in a string of length n, where τ is the number of maximal runs in the string [76].

In this chapter we will consider subsequences with bounded flexibility.

Definition 2. *Given a string s and an integer $\omega \geq 0$, an* ω-occurrence *of a string v in s is an occurrence $\langle i_0, i_1, \ldots i_{|v|-1}\rangle$ such that $0 \leq i_0 < i_1 < \cdots < i_{|v|-1} < |s|$ and $0 \leq i_{j+1} - i_j - 1 < \omega$ for all $0 \leq j < |v|$. Every substring of s that contains an ω-occurrence of v is called an* ω-realization *of v. A string v that has an ω-occurrence in s is called an* ω-subsequence *of s.*

Definition 3. *The ω-occurrence of a string v that starts at position j in s and that corresponds to the sequence of lexicographically smallest positions among all other ω-occurrences that start at j is called* greedy at j.

Given an ω-occurrence $\mathbf{i} = \langle i_0, i_1, \ldots, i_{|v|-1}\rangle$ of v in s, the *window* of $\mathbf{i}$ is the word $w_{\mathbf{i}} = s[i_k + 1 \ldots i_k + \omega]$. We extend s by the segment $s[|s|, |s| + 1, \ldots |s| + \omega - 1]$ filled with the extra character $\$ \notin \Sigma$, so that every ω-occurrence has a window. For any position j of s, the *set* $\mathcal{H}_j$ of windows of the ω-occurrences of v starting at j is called the *horizon* of v *at* j. The set of windows of all ω-occurrences of v in s is called the *panorama* $\mathcal{P}_v$ of v in s. We say that symbol $a \in \Sigma \cup \{\$\}$ is *visible* in $\mathcal{P}_v$ if there is at least one word in $\mathcal{P}_v$ that contains it. If $\omega = 1$, each horizon contains exactly one window, and the panorama cannot contain more than $|\Sigma| + 1$ total windows. To examine a more elaborate case, let:

$$\omega = 3,\ s = \texttt{ACCTATACGT\$\$\$},\ v = \texttt{ATAT},\ w = \texttt{ACT}\,.$$

Word v has ω-occurrences: $\mathbf{i_1} = \langle 0, 3, 4, 5\rangle$, $\mathbf{i_2} = \langle 0, 3, 6, 9\rangle$ and $\mathbf{i_3} = \langle 4, 5, 6, 9\rangle$. Word w has ω-occurrences: $\mathbf{j_1} = \langle 0, 1, 3\rangle$, $\mathbf{j_2} = \langle 0, 2, 3\rangle$, $\mathbf{j_3} = \langle 0, 2, 5\rangle$, $\mathbf{j_4} = \langle 4, 7, 9\rangle$ and $\mathbf{j_5} = \langle 6, 7, 9\rangle$. Therefore, $\mathcal{L}_v = \{0, 4\}$, $\mathcal{L}_w = \{0, 4, 6\}$, $\mathcal{R}_v = \{5, 9\}$ and $\mathcal{R}_w = \{3, 5, 9\}$. Word v has panorama $\mathcal{P}_v = \{\mathtt{ACG}, \$\$\$\}$, in particular, $\mathcal{H}_1 = \{\mathtt{ACG}, \$\$\$\}$ and $\mathcal{H}_5 = \{\$\$\$\}$. Word w has panorama $\mathcal{P}_w = \{\mathtt{ATA}, \mathtt{ACG}, \$\$\$\}$ with $\mathcal{H}_1 = \{\mathtt{ATA}, \mathtt{ACG}\}$, $\mathcal{H}_5 = \{\$\$\$\}$, $\mathcal{H}_7 = \{\$\$\$\}$. The greedy ω-occurrences of v are $\mathbf{i_1}$ and $\mathbf{i_3}$, those of w are $\mathbf{j_1}$, $\mathbf{j_4}$ and $\mathbf{j_5}$.

Note that the number of ω-occurrences of strings of length k starting at the same position in s is $O(\omega^{k-1})$, and the maximum number of ω-occurrences of a specific string v in s is $O(\omega^{|v|-1} \cdot |s|)$. This upper bound is tight, being attained by $v = \mathtt{A}^{|v|}$ in $s = \mathtt{A}^{|s|}$. The maximum number of distinct strings of length k that ω-occur in s is $O(\min(|\Sigma|^k, |s| \cdot \omega^{k-1}))$; this bound is attained by $\Sigma = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$, $s = (\mathtt{ACGT})^N$, $\omega = 4$, $N \gg k$. The number of greedy ω-occurrences of a specific string v in s is $O(|s|)$, and the maximum number of greedy ω-occurrences of strings of length k that start at the same position in s is $O(\min(|\Sigma|, \omega)^{k-1})$. For ease of notation, from now on we will use "occurrence" to mean an ω-occurrence, and we will indicate the value of ω by appending ω replicas of "\$" at the end of string s.

The ω-*complexity* of a string s is classically defined as function $f_s(\omega)$ that assigns to every integer $\omega \geq 0$ the number of distinct ω-subsequences of s [84]. The structure of ω-subsequences in general, and the ω-complexity in particular, are closely related to Fibonacci recursions [84]. A closed form for the tight upper bound of the ω-complexity of any string of length n can be derived using generating functions [84]. In this chapter, we aim at measuring the complexity of a string using higher-order constructs that relate to classes of positional equivalence, rather than by counting the number of subsequences or of ω-subsequences.

Definition 4 (Left equivalence)**.** *Two subsequences v and w are* left equivalent*, denoted $v \equiv_l w$, if $\mathcal{L}_v = \mathcal{L}_w$.*

We stipulate that strings never occurring in s are assigned to the class characterized by the empty list. We also assign the left list $\{0, 1, \ldots, |s|-1, |s|\}$ to $v = \varepsilon$ and $\{|s|\}$ to $v = \$$. With this proviso, the equivalence relation $\equiv_l$ imposed on string s is a partition of $\{\Sigma^* \cup \{\$\}\}$, containing at most $2^{|s|} + 2$ *left-equivalence classes*. The *right equivalence* relation $\equiv_r$ and its corresponding classes are defined symmetrically. The following properties are immediate from the definitions.

Property 1. *If $v \equiv_r w$, then $\mathcal{P}_v = \mathcal{P}_w$.*

Property 2. *The relation $\equiv_r$ is right-invariant, i.e., $v \equiv_r w$ implies $va \equiv_r wa \; \forall \; a \in \Sigma \cup \{\$\}$.*

Note that v and w can have the same panorama even though they do not have the same number of occurrences in s, and that $\mathcal{P}_v = \mathcal{P}_w$ may occur even if the relation $v \equiv_r w$ does not hold: for example, in $s = \texttt{ATCACGTCAC\$\$}$ we have $\mathcal{P}_{\texttt{AT}} = \mathcal{P}_{\texttt{GT}} = \{\texttt{CAC}\}$ even if AT and GT are not right-equivalent.

Definition 5 (Implication)**.** *We say that w* implicates *or* induces *v on s if, for every occurrence $\mathbf{i^1} = \langle i^1_0, i^1_1, \ldots, i^1_{|w|-1}\rangle$ of w, there is also an occurrence $\mathbf{i^2} = \langle i^2_1, i^2_2, \ldots, i^2_{|v|-1}\rangle$ of v such that $(i^1_0 = i^2_0)$ and $(i^1_{|w|-1} = i^2_{|v|-1})$.*

Implication is not symmetric, e.g., with $s = \texttt{ACAGTTT\$\$\$}$, $v = \texttt{AGT}$ and $w = \texttt{ACT}$, we have that w implicates v even though v does not implicate w.

Definition 6 (Equivalence)**.** *Two subsequences v and w of s are* equivalent*, denoted $v \equiv w$, if they implicate one another.*

We say that a class of the equivalence relation $\equiv$ is a *terminal class* if the list of its right occurrences is $\{|s|-1\}$. Every subsequence in such a class is called *terminal subsequence.*

Lemma 1. *If $v \equiv w$, then $\mathcal{P}_v = \mathcal{P}_w$; moreover, v and w have the same horizon structure.*

Proof. If $v \equiv w$ then $v \equiv_r w$, hence $\mathcal{P}_v = \mathcal{P}_w$. For a generic i, consider the set I_i of the occurrences of v starting at i: the occurrences of w that also start at i are precisely the occurrences implicating the occurrences of I_i, therefore w has a horizon at i that coincides with the one of v. □

Lemma 2. *The equivalence relation $\equiv$ is right-invariant.*

Proof. It is immediate that the generic occurrence of wa in s is implicated by at least one occurrence of va, and vice versa. Hence, $v \equiv w$ implies $va \equiv wa \; \forall \; a \in \Sigma$. □

Note that $v \equiv w \;\; \Rightarrow (\mathcal{L}_v = \mathcal{L}_w) \;\wedge\; (\mathcal{R}_v = \mathcal{R}_w)$, but the converse is not true. Consider the example $s = \texttt{ACATCATCATCT\$\$\$}$, $v = \texttt{AT}$, $w = \texttt{ACT}$, where $\mathcal{L}_v = \mathcal{L}_w = \{0, 2, 5, 8\}$ and $\mathcal{R}_v = \mathcal{R}_w = \{3, 6, 9, 11\}$: occurrence $\mathbf{i_1} = \langle 5, 6\rangle$ of v does not have a corresponding occurrence of w starting at position 5 and ending at position 6, and the occurrence $\mathbf{i_2} = \langle 5, 7, 9\rangle$ of w does not have a corresponding occurrence of v starting at position 5 and ending at position 9.

1.3 Special subsequences and the ω-suffix space

It is of interest to single out the subsequences of s that cannot be expanded without losing *support*, i.e., their number of ω-occurrences in s. The following definition may be considered an extension to subsequences of the one applied to substrings in [39].

Definition 7 (Special subsequence)**.** *A subsequence $v \in \Sigma^*$ occurring in s starting with left list $\mathcal{L}_v \neq \emptyset$ is a* special *subsequence of s if $\mathcal{L}_{va} \subset \mathcal{L}_v$ for every symbol $a \in \Sigma \cup \{\$\}$ visible from $\mathcal{P}_v$. A subsequence v is* non-special *if there is a symbol $a \in \Sigma \cup \{\$\}$ visible from $\mathcal{P}_v$ such that $\mathcal{L}_{va} = \mathcal{L}_v$.*

Note that, according to this definition, ε is a special subsequence. Special subsequences have the following properties.

Property 3. *Let $i_0, i_1, \ldots, i_{k-1}$ be the starting positions of v in s. Then v is a special subsequence if and only if there are two starting positions i_h and i_k such that no window in $\mathcal{H}_{i_h}$ shares a symbol with a window in $\mathcal{H}_{i_k}$.*

Property 4. *If av is a special subsequence and $a \in \Sigma$, then the suffix v of av is a special subsequence.*

Property 5. *If v is a special subsequence, then such is also any $w \equiv v$.*

Following is the dual notion of that of a special subsequence.

Definition 8 (Antispecial subsequence)**.** *A subsequence v of s is* antispecial *if any extension va of v in s, $a \in \Sigma \cup \{\$\}$, results in $va \equiv_l v$.*

Therefore, a subsequence is antispecial if and only if every symbol with which it can be extended in s appears in every horizon. Notice that a subsequence v that is extensible in s in only one way, or such that $|\mathcal{L}_v| = 1$, is necessarily antispecial, but an antispecial subsequence can have any support in general. It is also easy to observe that the extensions of an antispecial subsequence, its prefixes and its suffixes are not necessarily antispecial. Finally, if v is antispecial, then every sequence $w \equiv v$ is antispecial.

The definition of special subsequence embodies a criterion to build all the $\equiv_l$ and $\equiv$ classes of a string s. Assuming that all the occurrences $\mathbf{i_0}, \mathbf{i_1}, \ldots, \mathbf{i_{k-1}}$ of a subsequence v in s have been found, we determine $\mathcal{L}_v = \{i_0, i_1, \ldots, i_{h-1}\}$ and then organize the windows in groups $\mathcal{H}_0, \mathcal{H}_1, \ldots, \mathcal{H}_{h-1}$ related to the same starting position of the occurrences: every symbol $a \in \Sigma \cup \{\$\}$ appearing in at least one window of the panorama $\mathcal{P}_v$ signals the occurrence of sequence va in s, that can be linked to v by a directed arc labeled by a, establishing a parent-child relationship between the sequences. If symbol a appears in at least one window of each group of v, then $va \equiv_l v$, otherwise va belongs to a new $\equiv_l$ class identified by $\mathcal{L}_{av} \subset \mathcal{L}_v$. If no child of

v belongs to the same class as v, then v is special, and if va and wb, where $a \neq b \in \Sigma$ and $v, w \in \Sigma^*$, have the same left and right lists, they belong to the same $\equiv$ class.

Like standard common subsequences, also those considered here are susceptible to a natural geometric representation. Let $\delta_0, \delta_1, \ldots, \delta_{d-1}$ be the positions of symbol $a \in \Sigma$ in s. Align the suffixes $s[\delta_i \ldots |s| - 1] \; \forall \; 0 \leq i < d$ along the d coordinate axes of a multidimensional grid, such that $s[\delta_i + k]$ occupies position $k+1$ along coordinate δ_i. This space, denoted $\Psi_a(s)$, will be called the ω-*suffix space* of s associated with character a. With the convention that the origin matches any character of Σ, we mark a *matching point* (in what follows often referred to simply as a *point* or *match*) in this space at every cell $\mathbf{X} = [X_0, X_1, \ldots, X_{d-1}]$ of the grid such that, $\forall \; 0 \leq i < d$, $(s[\delta_i + X_i - 1] = b) \vee (X_i = 0)$, $b \in \Sigma$. Next, we define a partial order on the points by using a strict-dominance criterion: $\mathbf{X} < \mathbf{Y}$ iff $X_0 < Y_0, X_1 < Y_1, \ldots, X_{d-1} < Y_{d-1}$.

A greedy ω-subsequence corresponds to a *chain* in this partial order, such that, for each pair of consecutive points $\mathbf{X}$ and $\mathbf{Y}$, we have $\mathbf{X} < \mathbf{Y}$ and for no point $\mathbf{Z}$ we have $\mathbf{X} < \mathbf{Z} < \mathbf{Y}$. To connect all chains related to some greedy ω-subsequence, we start at the origin and connect matches in succession under the ω constraint, and in such a way that whenever an arc is established between points $\mathbf{X}$ and $\mathbf{Y}$, then for no point $\mathbf{Z}$ we have $\mathbf{X} < \mathbf{Z} < \mathbf{Y}$. Except for the fact that here we direct the arcs from the lower point to the higher one, this process results in a partial Hasse diagram for the partially-ordered set of points [66], that is, the portion of the diagram that is compatible with the ω constraint. Still, there are greedy ω-subsequences that are not captured in this process.

The simple construction that we now proceed to describe traces *all* the ω-sequences of s, resulting in what constitutes an expansion of the constrained Hasse diagram above. The space $\Psi_a(s)$ sets the natural stage also for such construction, which starts at point $\mathbf{1} = [1, 1, \ldots, 1]$ and proceeds according to the following rule. Assume that, at the generic iteration, we are at point $\mathbf{X} = [X_0, X_1, \ldots, X_{d-1}]$, and let $Y_0^c, Y_1^c, \ldots, Y_{d-1}^c$

be the closest matches of character $c \in \Sigma$ following $\mathbf{X}$ in the partial order and falling within an interval of ω on every axis: in the next iteration, we move to all such points and resume the process. Is it apparent that this construction explores only a subset of all matching points in $\Psi_a(s)$, and that, in a generic space with d dimensions, it proceeds monotonically within an hyperpyramid with vertex at point $\mathbf{1}$, axis along the line passing through the points with equal coordinates, and edges identified by the d lines passing through the points $[1+\omega, 2, 2, \ldots, 2]$, $[2, 1+\omega, 2, \ldots, 2]$, $\ldots$, $[2, 2, \ldots, 1+\omega]$ and through the vertex. This process mimics the construction of a trie with analogies to the *inexact suffix tree* [34]; for unbounded ω, the resulting graph is seen to incorporate the Hasse diagram of the poset of matches.

Let V be the set of points in space $\Psi_a(s)$ that are visited by the algorithm just described, and let A be the set of arcs, oriented and labeled by symbols of Σ, that indicate the extensions of each point of V carried out by the algorithm. Graph $G_a(s) = (V, A)$ is called the *ω-suffix graph* induced by symbol a on s.

Lemma 3. *The points of $G_a(s)$ represent all and only the classes of equivalence relation $\equiv$ among the greedy ω-subsequences of s that start by symbol a.*

Proof. Strings that share the same starting and ending positions in all their greedy ω-occurrences in s are clearly projected to the same point of $\Psi_a(s)$. That these points identify all the equivalence classes of the $\equiv$ relation derives from the fact that the procedure generates all the subsequences that have a greedy ω-occurrence in s. Note that the left list of the class associated with point $\mathbf{X} = [X_0, X_1, \ldots, X_{d-1}]$ is given by $\mathcal{L} = \{\delta_i \ : \ X_i \neq 0, \ 0 \leq i < d\}$, and that the right list is given by $\mathcal{R} = \{\delta_i + X_i - 1 \ : \ X_i \neq 0, \ 0 \leq i < d\}$. □

$\Psi_a(s)$ may have a very high number of dimensions, but it contains subspaces of smaller dimensionality.

Definition 9 (Subspace of $\Psi_a(s)$)**.** *Let $\delta_0, \delta_1, \ldots, \delta_{d-1}$ be the list of occurrences of*

symbol a in s. The subspace of $\Psi_a(s)$ associated with the distinct coordinates $\delta_{i_0}, \delta_{i_1}, \ldots, \delta_{i_{k-1}}$, *is the set of points in $\Psi_a(s)$ having non-null values only along dimensions $\delta_{i_0}, \delta_{i_1}, \ldots, \delta_{i_{k-1}}$.*

Strings belonging to the same class under $\equiv_l$ are projected to paths of $G_a(s)$ ending at points located in the same subspace of $\Psi_a(s)$. In particular, class $\mathcal{L}_a$ associated with point $\mathbf{1}$ consists of points with exactly d non-null coordinates, the class formed by strings that never occur in s corresponds to point $\mathbf{0} = [0, 0, \ldots, 0]$ having zero non-null coordinates, and the transition from the string v of a class that comprises coordinate δ to the string va of a class devoid of coordinate δ corresponds to the connection of the last point associated with the $\equiv$ class of v to a point with a null value along δ. A special subsequence is associated with a path of the graph ending at a point $\mathbf{X}$ from which it is only possible to connect to points located in subspaces with fewer dimensions than $\mathbf{X}$; an antispecial subsequence, on the other hand, corresponds to a path ending at a point $\mathbf{Y}$ which is able to connect only to points with the same non-null coordinates as $\mathbf{Y}$. From this geometric interpretation it can be seen that speciality, antispeciality, and the way in which the support is reduced by extension, are properties common to all the subsequences belonging to the same equivalence class of relation $\equiv$, as was noted earlier.

1.4 Core equivalence classes

Even though graph $G_a(s)$ visits an exponential number of points across a large number of subspaces, its structure exhibits a high degree of redundancy: for instance, if two points have the same value along some coordinate δ (i.e., they lie in the same hyperplane orthogonal to coordinate δ), their extension by every possible symbol leads to points that still have the same value along coordinate δ. More generally, the points of $G_a(s)$ lying in each subspace S_k with $k < d$ dimensions belong to the graph that our algorithm would create if it were executed within the sole subspace S_k. It is seen,

however, that the entire population of the classes of $\equiv$ can be described using only a limited number of representatives. The main reason for this is the fact, that along each coordinate axis of $\Psi_a(s)$ there is at least one point for each discrete value within that coordinate range: we use this observation to derive a small subset of points with the property that the structure of all classes may be inferred from them.

Consider a map that assigns to every value $k > 1$ along every coordinate δ_i of space $\Psi_a(s)$ exactly one point $\mathbf{X} = [X_0, X_1, \ldots, X_{d-1}]$ of graph $G_a(s)$, such that the i-th coordinate of $\mathbf{X}$ is equal to k. There are many ways to choose such points. To fix the ideas, we will select the *diagonal* map ϱ such that, for any dimension $\delta_j \neq \delta_i$, X_j is the largest possible value not greater than k. The points identified by ϱ are called *core points*, and they form a set $R_a = \{\mathbf{X^0}, \mathbf{X^1}, \ldots, \mathbf{X^{r-1}}\}$; the equivalence classes of relation $\equiv$ associated with these points are called *core classes*; the points (and the associated classes) that are not included in R_a are called *residual points*. Core equivalence classes in topological order describe in a compact way the whole structure of $G_a(s)\ \forall\ a \in \Sigma$, in that they are enough to retrieve all other points.

Lemma 4. *Pair (R_a, A_a) formed by core points and by the arcs individually leaving core points in $G_a(s)$ enables to reconstruct the connection of any residual point, without knowledge of the original string s.*

Proof. Let $\mathbf{Y} = [Y_0, Y_1, \ldots, Y_{d-1}]$ be the generic residual point. By assumption, we know the group formed by the not necessarily distinct core points $A = \{\mathbf{X_0}, \mathbf{X_1}, \ldots, \mathbf{X_{d-1}} : Y_i = X_{i,i}\ \forall\ 0 \leq i < d\}$. Consider a symbol $a \in \Sigma$, and suppose that the arcs labeled by a leaving each point of A lead, respectively, to the not necessarily distinct points $B = \{\mathbf{Z_0}, \mathbf{Z_1}, \ldots, \mathbf{Z_{d-1}}\}$: each point $\mathbf{Z_i}$ indicates the closest occurrence of a along coordinate i. Thus, collecting the individual values yields the match to which $\mathbf{Y}$ is to be connected under a transition labeled by a. □

By way of illustration, consider the example in Figure 1: the residual point $[4, 6]$

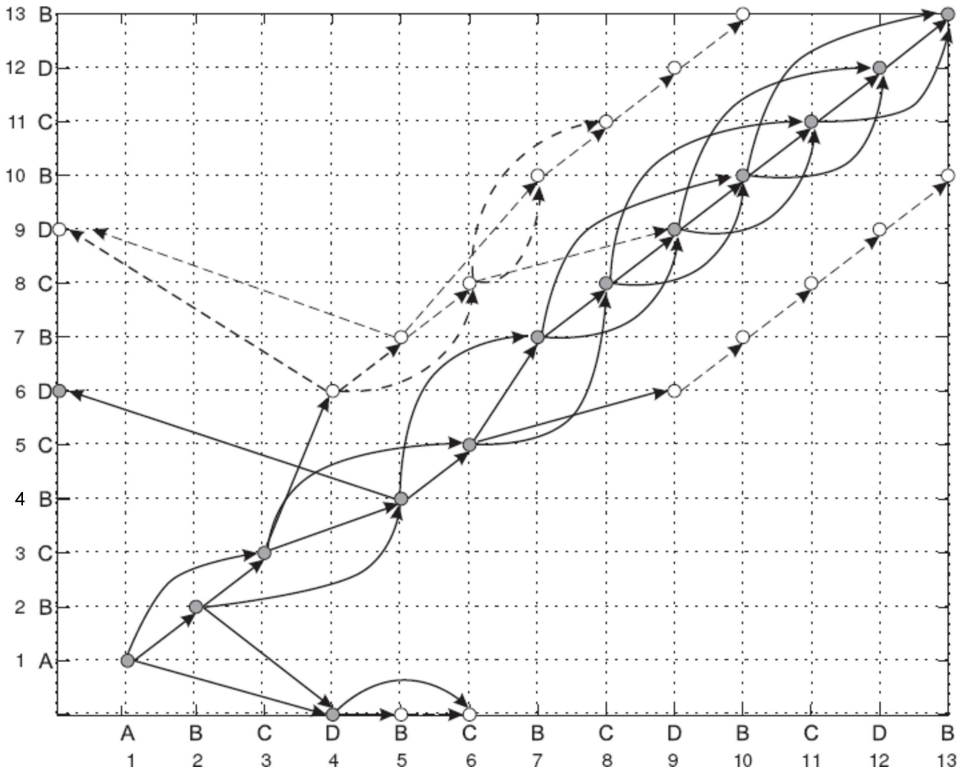

Figure 1: Core points and residual points for a case with $\omega = 3$. Core points and their outbound arcs are shown in dark gray; residual points and part of their outbound arcs are shown in light gray and dashed, respectively.

can be reconstructed by inspecting the outbound connections of the core point $[3, 3]$. Because $[4, 6]$ is residual, there are core points $[4, 0]$ and $[0, 6]$ that have the same values as $[4, 6]$, respectively along the horizontal and vertical axis. In particular, the extension of $[4, 0]$ by symbol `B` leads to point $[5, 0]$, and the extension of $[0, 6]$ by the same symbol leads to $[0, 7]$: therefore, it must be that the extension of $[4, 6]$ by `B` leads to point $[5, 7]$. The same procedure can be applied to predict the points to which $[4, 6]$ connects using the other symbols of Σ.

Lemma 4 guarantees that the knowledge of the core classes of equivalence relation $\equiv$, and of at most $|\Sigma|$ residual classes for each of them, suffices to determine all the classes of relations $\equiv$ and $\equiv_l$ and all the subsequences belonging to them; notice, in particular, that we can reconstruct the lists $\mathcal{L}$ and $\mathcal{R}$ of each $\equiv$ class.

Lemma 5. *The number of core classes induced by $\equiv$ on a string s is $O(|\Sigma| \cdot |s|^2)$ and*

$\Omega(|s|)$. *The graphs* $G_a(s)\ \forall\ a \in \Sigma$ *can be recovered from* $O(|\Sigma|^2 \cdot |s|^2)$ *core points and arcs.*

Proof. For arbitrary $a \in \Sigma$, every core point in $G_a(s)$ is associated with at least one value along the i-th coordinate: therefore, we can assign a name to each of these points according to one of the values they are associated with by function ϱ. There are at most $|s|$ names for each of the d coordinates: under the most restrictive hypothesis, each point will be associated with only one value along one coordinate, whence the number of core points in $G_a(s)$ cannot exceed $d \cdot |s|$. Since this holds for each $a \in \Sigma$, and $d \leq |s|$, the number of core classes cannot exceed $|\Sigma| \cdot |s|^2$. The lower bound follows from the immediate observation that if a is the first character of s, then there is a path in (R_a, A_a) that spells out precisely s. By Lemma 4, to reconstruct graph $G_a(s)$ it suffices to know the $|s|^2$ core points of space $\Psi_a(s)$ and their outbound arcs. Each such point has at most $|\Sigma|$ outbound arcs, and the spaces to be considered are at most $|\Sigma|$. Hence the overall number of points and arcs required to reconstruct all graphs $G_a(s)\ \forall\ a \in \Sigma$ is $O(|\Sigma|^2 \cdot |s|^2)$. □

The number of core points can be much lower than the upper bound if ϱ chooses points associated each with multiple values along many dimensions. It seems interesting that the upper bound to the number of core classes in the above lemma does not depend on ω, and in particular that it does not change while ω shifts from 1 to greater values. When $\omega = 1$ no point of $G_a(s)$ is residual; since points correspond, in this case, to distinct substrings of s, the number of core classes is still $O(|\Sigma| \cdot |s|^2)$. This independence may seem surprising at first sight, since the complexity of the structure of equivalence classes presents a clear discontinuity at $\omega = 1$. Indeed, when $\omega = 1$ the lists $\mathcal{L}$ of all child subspaces are *partitions* of the $\mathcal{L}$ lists of their parent spaces, because each subspace is reachable, through a different symbol of the alphabet, from the sole point associated with the special sequences of the parent space. When $\omega > 1$ the increased width of the window does not force any more the construction to *either*

remain within a subspace *or* to leave it, but it let it exploit both of these possibilities by carrying out extensions by different symbols. Therefore, the points associated with special sequences can be more than one for each subspace, and the exit from a subspace can be possible also from points that are not associated with special sequences. Consequently, it is not necessary for the lists $\mathcal{L}$ associated with each of the subspaces reachable from a space S to be partitions of the list $\mathcal{L}$ of space S: it suffices for them to be *subsets*.

1.5 Structure in artificial strings

It is natural to ask in what ways do suffix graphs expose the structure of strings. In particular, a basic question is whether any of these features can separate random from ordered strings, and be used as a complexity parameter to classify and compare strings. We describe here the results of a controlled experiment on 10 artificial strings, expected to possess different degrees of internal structure.

- CONSTANT - The constant string $(\mathtt{0})^{200}$.
- PERIODIC - The periodic string $(\mathtt{0123456789})^{20}$.
- BLOCK - The block string $(\bullet_{i=0}^{9} i^5)^4$, where $\bullet$ denotes concatenation.
- QPERIODIC - A random quasiperiodic string, with quasiperiod `15374628091537`. Recall that a string s is *quasiperiodic* if there is a string $w \neq s$ such that the occurrences of w in s completely cover s, that is if every position in s belongs to at least one occurrence of the *quasiperiod* w of s [15]. The string that we use here is built by iteratively choosing, with equal probability, whether to concatenate the quasiperiod or to partially overlap it to the current prefix.
- RANDOM - A string emitted by a pseudo-random source assigning equal probability to all symbols in Σ.

- π - The truncation to the first 200 decimal digits of the irrational constant π.
- ϕ - The truncation to the first 200 decimal digits of the golden ratio $\phi = (1 + \sqrt{5})/2$.
- CHAMP - The truncation to the first 200 decimal digits of the Champernowne constant C_{10}. Recall that the Champernowne number C_b on base b is represented by concatenating to "0." the infinite string s consisting of the concatenation of the base-b representations of the natural numbers, in increasing order. String s is *disjunctive*, i.e. it has a number of distinct substrings of length $i > 0$ equal to b^i. For particular choices of b (like 10) it is also *normal*, i.e. all strings of the same length occur in s asymptotically with the same frequency: in this case, the probability of finding a string w in a given portion of s equals what we would expect in a random string.
- ERDOS - The truncation to the first 200 decimal digits of the Copeland-Erdös constant. Recall that this number is represented by concatenating to "0." the base-10 representations of all prime numbers, in increasing order. This number is normal and disjunctive.
- PROTEIN - The primary sequence of a single-strand binding protein from *Clostridium botulinum* (entry `ACA57568` in NCBI). The alphabet has been reduced from 20 to 10 symbols by recoding each amino acid according to its Lifson-Sander value that measures the conformational preference of an amino acid for parallel beta-strand secondary structure [149]. The scale of possible values has been uniformly divided into bins of equal size, and all amino acids falling in the same bin have been assigned the same symbol.

All strings are defined on alphabet $\Sigma = \{0, 1, \ldots, 9\}$, have length 200, and are scanned from left to right during the construction of their suffix graphs.

Recall that two key properties of a point in a suffix graph are its label and its dimensionality; in particular, we will call *internal* the arcs that connect points embedded in the same subspace, and *external* the arcs associated with extensions that cause a loss of dimensionality; as for points, a key additional property of an arc is its label. The simplest measures that can be collected from a suffix graph are perhaps those listed below.

- *Measures on points* — These include the total number of points (in particular, total number of special points, antispecial points and normal points), and the total number of points with a given label and dimensionality (again divided into special points, antispecial points and normal points).

- *Measures on arcs* — These consist of the total number of arcs (in particular, internal and external arcs), the number of internal arcs with a given label and lying in a subspace with a given number of dimensions, the number of external arcs removing a given number of dimensions and having a given label, and the number of external arcs directed from a subspace with d_1 dimensions to a subspace with $d_2 < d_1$ dimensions.

- *Associative measures* — Associations among symbols are measured by counting the number of points with a given label and a given fan-out, the number of points having both symbol a and b in their fan-out, the number of directed arcs from symbol a to symbol b, and the number of external arcs with label b departing from a point with label a.

In the rest of this section we make the further simplification of considering the sum of each feature over the suffix graphs associated with all symbols of Σ; in other words, if $f_i(s, a)$ represents feature i measured on the suffix graph of string s associated with symbol $a \in \Sigma$, we actually study $f_i(s) = \sum_{a \in \Sigma} f_i(s, a)$. We do not normalize these

counts, since all strings in our dataset have the same length and are defined on the same alphabet[2].

We focus our analysis on $\omega = 4$. The total number of points reveals remarkable differences among the strings above. In particular, the dataset seems to be divided into three groups (Figure 2, top-left): CONSTANT, PERIODIC, BLOCK and QPERIODIC have a markedly lower number of points than RANDOM, while CHAMP, ERDOS and PROTEIN are greater; π and ϕ are indistinguishable from RANDOM: this seems to fall in line with the fact that these sequences have passed the usual statistical tests for randomness. The more detailed graphs displayed in Figure 2 make this classification even stronger, showing that ϕ has less special and normal points than RANDOM; even π seems to be different from RANDOM when special and antispecial points are taken into account, but these differences are weak. By analyzing all graphs, it seems therefore reasonable to group the dataset into the following three classes: (CONSTANT, PERIODIC, BLOCK, QPERIODIC), (RANDOM, π, ϕ), (CHAMP, ERDOS, PROTEIN). Similar conclusions can be drawn from the counts of the number of arcs: in this case, ϕ displays a lower number of external arcs than RANDOM, and ERDOS shows a significantly greater number of arcs than CHAMP and PROTEIN.

The correlation between label and dimensionality in points reveals further structure. Zooming into the bars of Figure 2, strong differences appear in the matrices traced by different strings (Figures 3 and 4). In these and all subsequent matrices, numbers are represented as uniform levels of intensity, increasing from dark to bright[3]. As expected, the graph of CONSTANT spans all subspaces, since it consists of a series of nodes with unitary fan-out, with each node in a different subspace; all points are

[2]Note, however, that not all strings in our dataset have the same number of occurrences of each symbol.

[3]Intensity is rescaled in each map to encompass its specific range of variation, therefore the same intensity can be associated with different numbers in different matrices.

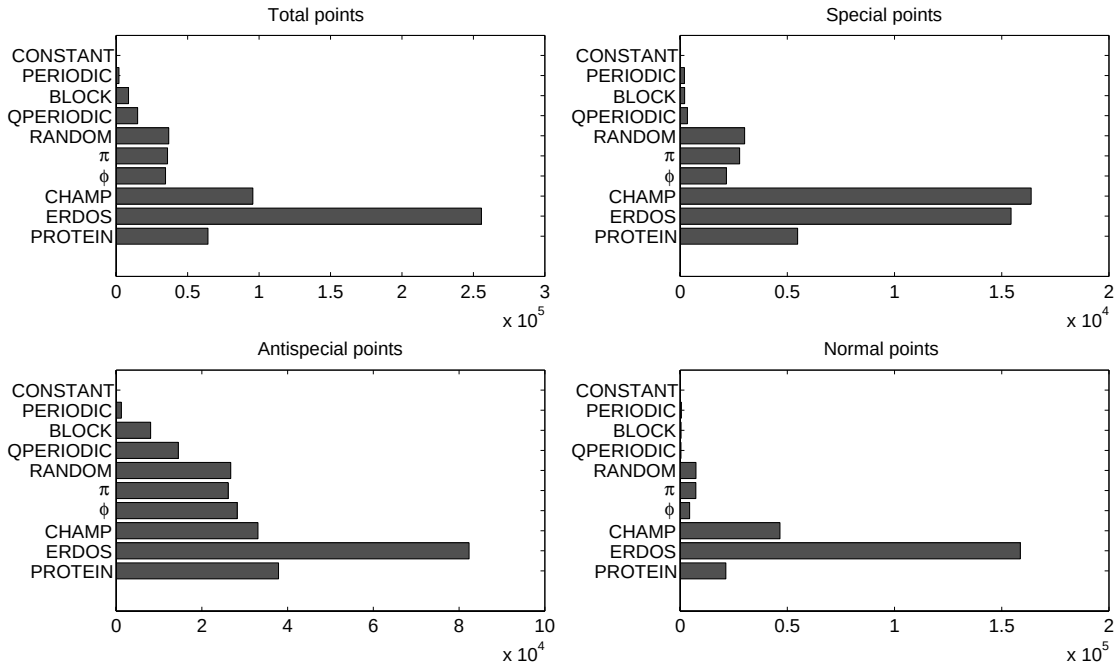

Figure 2: Total number of points in artificial strings ($\omega = 4$).

special, since the support undergoes a unit decrease with each extension. By inspecting the numerical values in the matrices of Periodic, we see that each suffix graph consists, disregarding border effects, of 20 filaments of 10 points each, in which each filament has exactly one special point, 3 normal points and 6 antispecial points. A regular pattern appears also in Block and, notably, in QPeriodic, even though the latter has been generated by a random process. Random, π and ϕ exhibit still a similar overall matrix, but differences in specific cells appear in the matrices of special points. The shapes of Champ and Erdos turn out to be completely different from each other and from those of all other strings, exhibiting each a peculiar pattern. Protein is different from all other strings as well, and it shows some similarities with Erdos. Analogous conclusions can be drawn by plotting the total number of points embedded in each dimensionality: to limit clutter, the curves of QPeriodic (a decreasing trend tending towards Periodic) and Block (approximately constant around 10^3) are not displayed in Figure 5, and Periodic and Constant are just

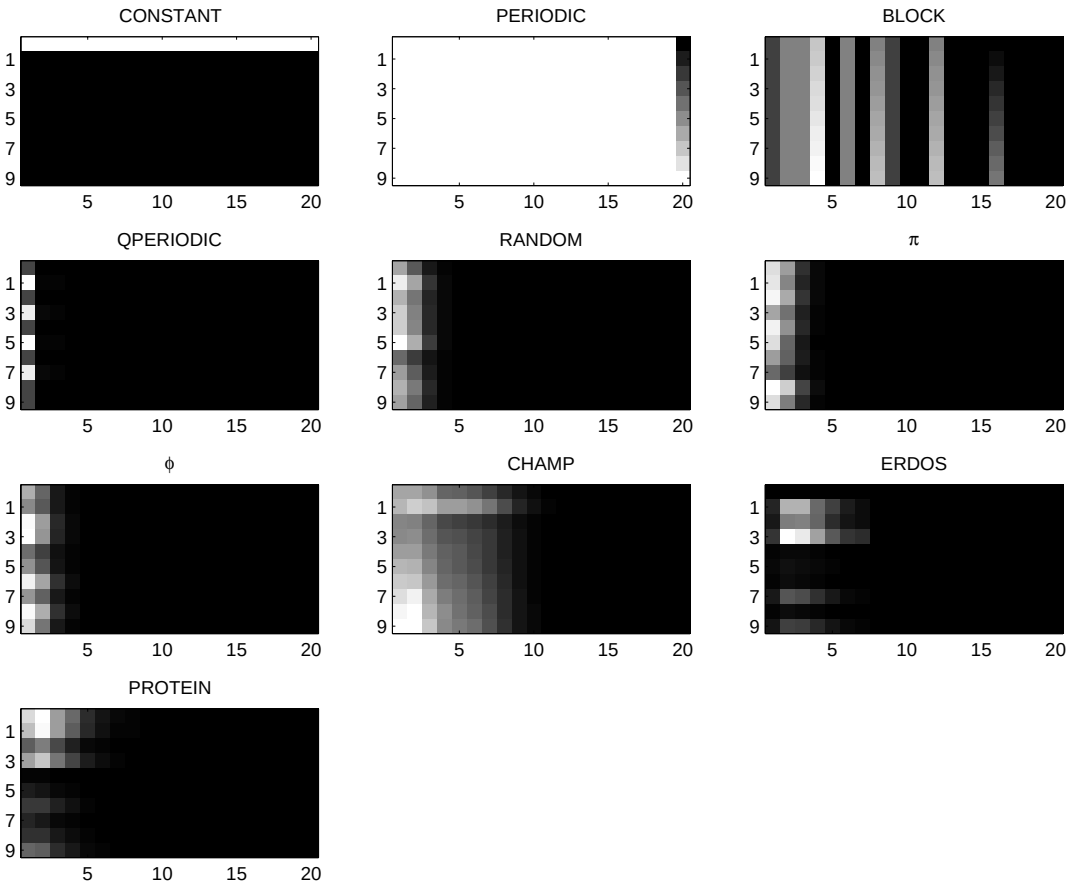

Figure 3: Total number of points with a given label (rows) and a given dimensionality (columns). $\omega = 4$.

sketched. The distribution of points over labels, the number of external arcs that connect subspaces of different dimensionality, and most of the measures on arcs suggest similar conclusions, confirming the proposed classification, highlighting similarities within each group and marked differences among groups. The shape of each map seems to be a peculiar signature of the corresponding string.

Further structure can be grasped by looking at associative measures: plotting the number of directed arcs between every pair of symbols (Figure 6), PERIODIC, BLOCK and CHAMP display a strongly banded trend; QPERIODIC appears to be organized in a highly-ordered checkerboard, despite the randomness of the process that produced it; ERDOS and PROTEIN display a clear horizontal band in the top part of their matrices; on the other hand, no differences seem to distinguish RANDOM, π and ϕ

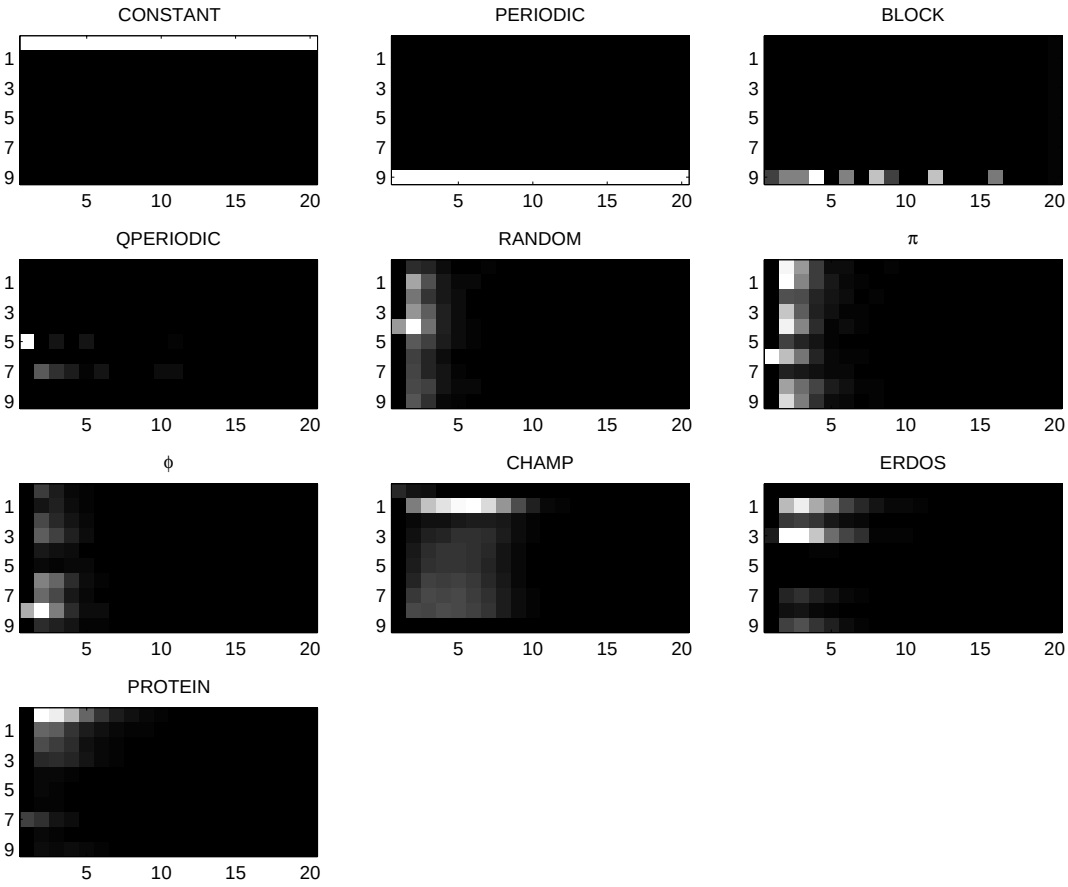

Figure 4: Number of special points with a given label (rows) and a given dimensionality (columns). $\omega = 4$.

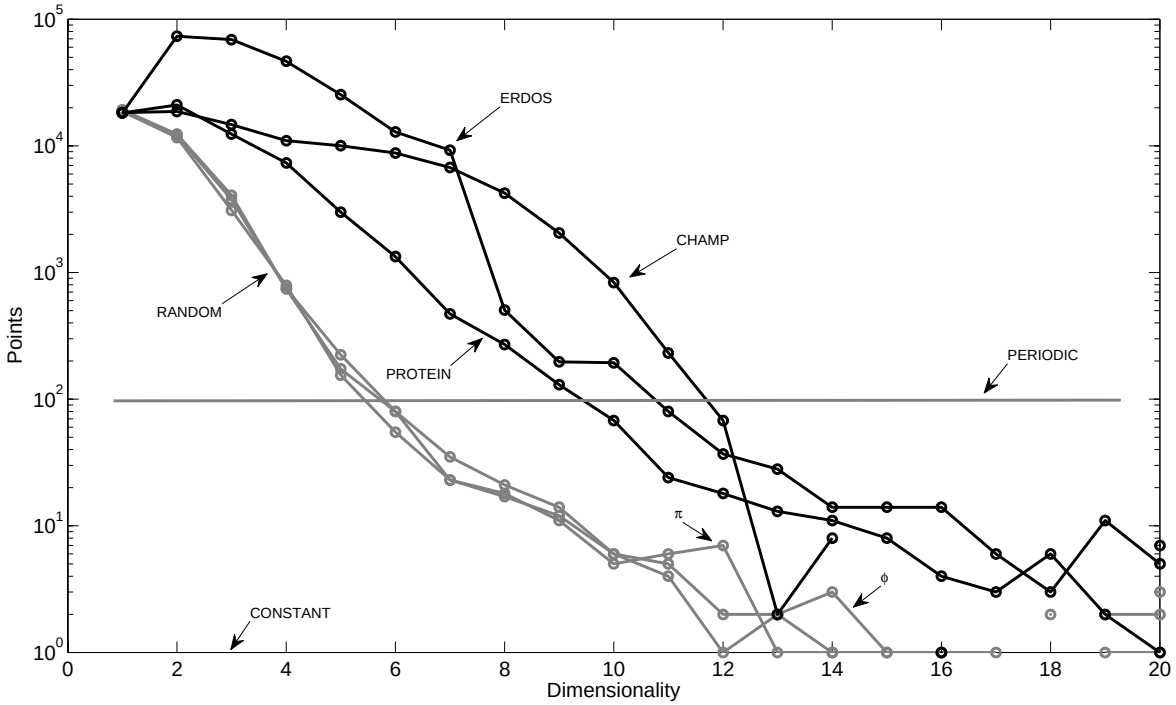

Figure 5: Total number of points embedded in subspaces of a given dimensionality ($\omega = 4$).

among each other. We obtain a similar classification if we plot the number of points with a given label and a given fan-out, the number of points having any pair of symbols in their fan-out, and if we consider the number of external arcs with label b that depart from a point with label a. Notably, in the latter case both π and ϕ display a shape that is clearly different from RANDOM (Figure 7). All these maps confirm the proposed classification, and also suggest that there is some similarity between CHAMP and the group of PERIODIC, while there are some strong differences between this group and QPERIODIC.

A full investigation of the effects of ω on the previous measures is outside the scope of this section. Preliminary results show that, in the range $2 \leq \omega \leq 8$, all maps tend to be similar to those already described. Differences among strings become generally fainter when ω decreases; when ω increases, the pattern of connectivity between symbols and the distribution of special points point out stronger differences among π, ϕ

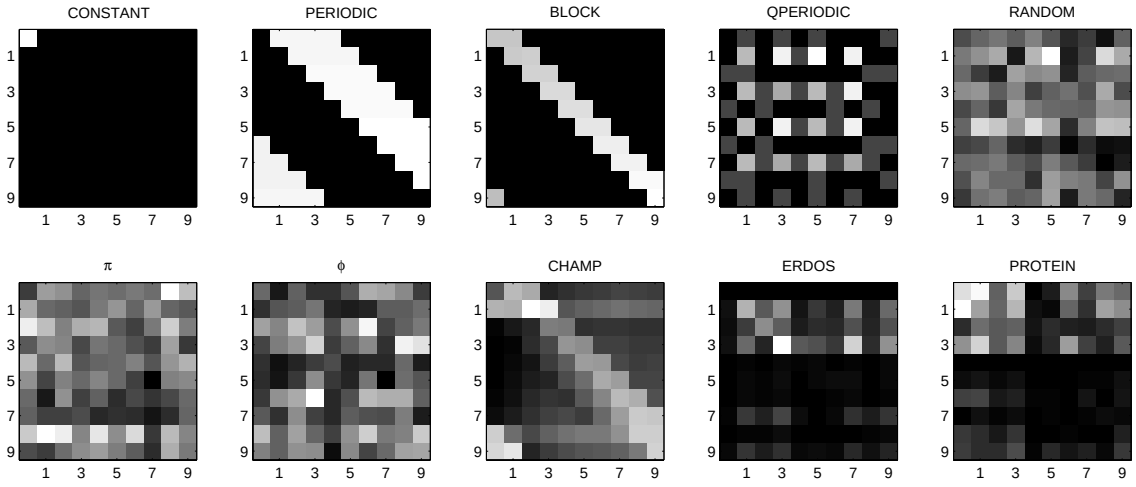

Figure 6: Number of arcs connecting every pair of symbols (rows: source symbol; columns: destination symbol). $\omega = 4$.

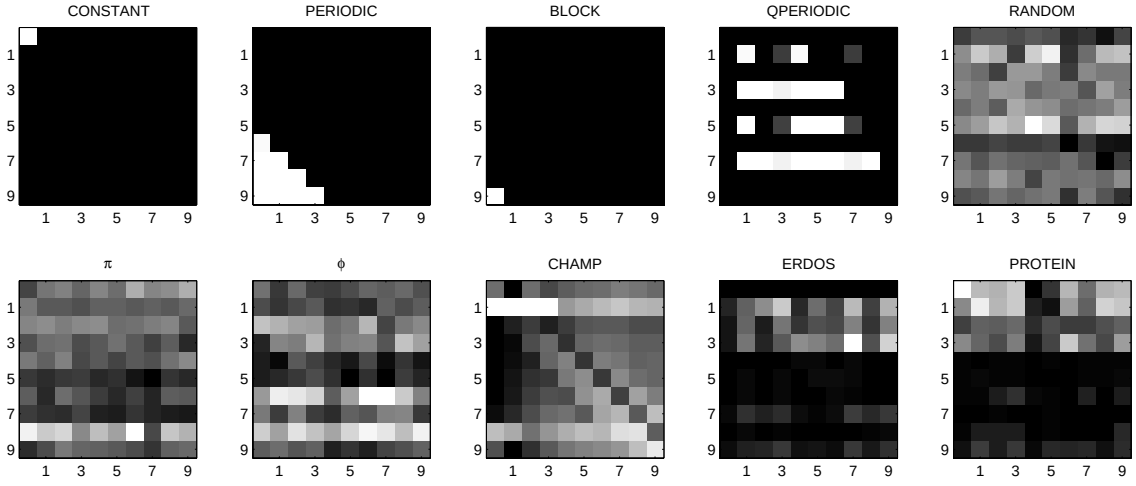

Figure 7: Number of external arcs with label b (columns) departing from points with label a (rows). $\omega = 4$.

and RANDOM. At high ω, the Euclidean distance of points from the origin of the space starts to reveal regular, differentiated profiles (Figure 8). When $\omega = 2$ the total number of points in PERIODIC, BLOCK and QPERIODIC becomes greater than RANDOM and comparable to CHAMP; in particular, CHAMP becomes significantly smaller than ERDOS and PROTEIN. Bar charts similar to Figure 2 seem in agreement with the proposed classification, but they further suggest that CONSTANT is significantly different from the other simple strings in its class. When $\omega = 8$ (Figure 9), CHAMP is still smaller than ERDOS and PROTEIN, but the group of RANDOM, π and ϕ displays a larger number of points and arcs than CHAMP, CONSTANT, PERIODIC and BLOCK. Now QPERIODIC can be separated from the other simple strings, since it assumes values that are more similar to RANDOM than to its own class. Notably, π exhibits the largest values in its class, surpassing CHAMP. These changes highlight that ω has a key role in determining the total number of points in the space, while preliminary analyses suggest that it does not induce significant alterations to the shape of the matrices.

1.6 Structure in polypeptides

Motivated by the high degree of structure revealed by suffix graphs in artificial strings, in the rest of this chapter we set up a battery of experiments to study the properties of suffix graphs generated by *amino acid strings* and by their random permutations. Natural compositional measures that we will consider are, again, the number of points (special, antispecial, normal, terminal), the number of arcs (internal and external) and the number of subsequences (special, antispecial, normal, terminal) at different values of ω. The following sections will aggregate and systematize over one million data points. This is probably the first time in which the structure and randomness of polypeptides is assessed at this scale in terms of the vocabulary of their constrained *subsequences* rather than their substrings.

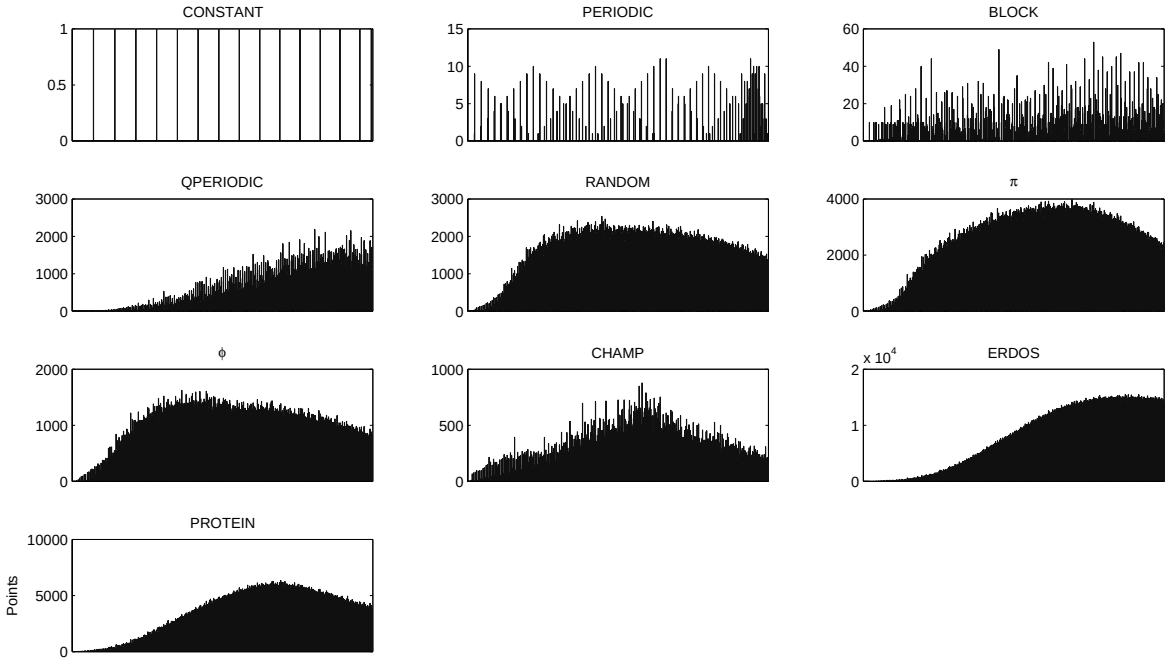

Figure 8: Total number of points at a given distance from the origin of the suffix space. The horizontal axis depicts interval $[0, 200]$ divided in bins of length 0.2. $\omega = 8$.

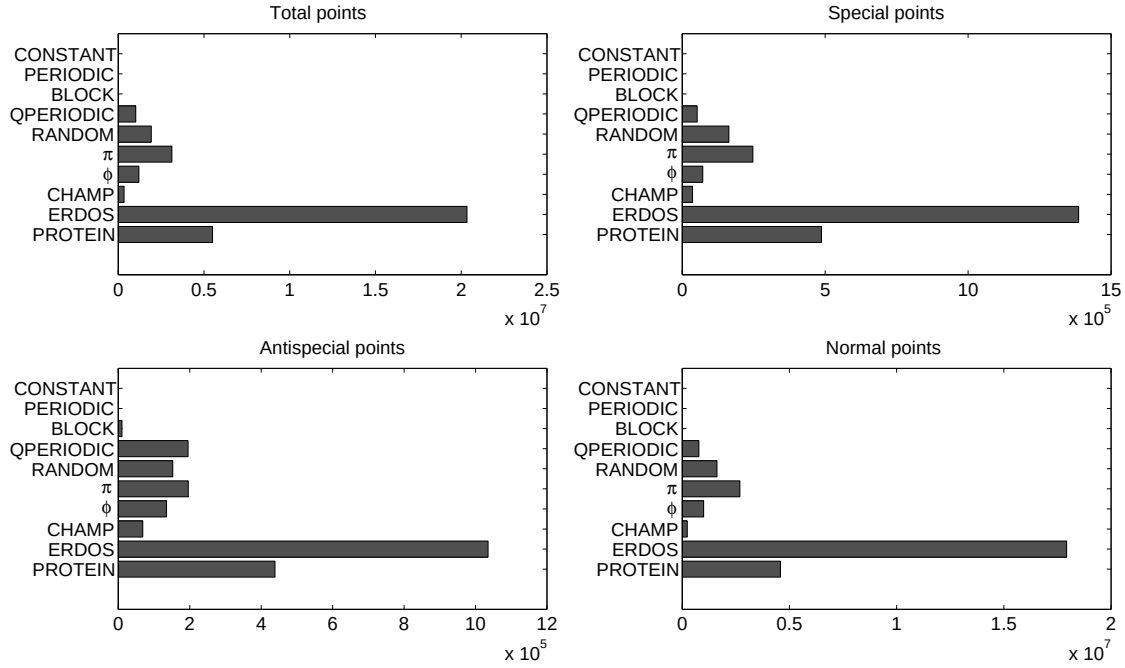

Figure 9: Total number of points at $\omega = 8$.

Many previous investigations [89, 94, 113, 173, 174] have recoded the original amino acid strings with reduced alphabets that incorporate physico-chemical information. The large number of such scales published to date [85], and the lack of a standard methodology to perform such recoding, haunted our preliminary experiments with parameters that made our analyses depend on their fine-tuning. Amino acid similarities are also known not to be universal: at different positions of a protein, different sets of amino acids or of amino acid substrings can more likely substitute for one another, making a fixed substitution scheme less biochemically significant [71]. The need to make our results as general as possible led us to analyze polypeptides encoded in the original alphabet of amino acids.

Most proteins consist of modular subunits (called *domains*) whose spatial conformation and function are thought to be independent of other parts of the protein. A limited number of highly similar domains are seen to occur in all known proteins, both as parts of larger multidomain structures and by themselves [112, 118, 136], suggesting that they are remnants of ancient functional polypeptides that have been assembled by evolution to produce the combinatorial variety of structures and functions that appear in modern proteins. The modular nature of domains makes them better candidates than whole proteins for investigating regularities and patterns, since the concatenation of different domains could be a source of noise. The sequential compactness and the moderate length of domains make them preferable to secondary structures, supersecondary structures, or motifs, that are typically shorter and non-contiguous subsequences.

The SCOP database [112] is a comprehensive ordering of all protein domains of known structure according to their evolutionary, structural and functional similarity. The basic classification unit is the domain, which is put at the leaves of a tree with three more hierarchical levels: *families*, containing domains with a common evolutionary origin as testified by high sequence similarity or highly similar function and

structure; *superfamilies*, containing domains with low sequence similiarity but sharing structural and functional features that suggest a common evolutionary origin; *folds*, containing domains with a specific set of major secondary structures, a specific configuration of these structures in space, and a specific connection pattern; and *classes*, containing domains that share the same frequency of secondary structures (e.g. domains in which the large majority of secondary structures are α-helices). This classification is manually curated by biologists.

We elect a subset of 148 SCOP domains as our main dataset: we will refer to this dataset as D_1 in what follows. We choose domains to span two different classes (1 and 2), two different folds per class (1.1, 1.8, and 2.1, 2.2), two different superfamilies per fold (1.1.1, 1.1.2, 1.8.1, 1.8.4, and 2.1.1, 2.1.2, 2.2.2, 2.2.3), and two different families per superfamily[4] (1.1.1.3, 1.1.1.4, 1.1.2.1, 1.1.2.2, 1.8.1.1, 1.8.1.2, 1.8.4.5, 1.8.4.6, and 2.1.1.1, 2.1.1.2, 2.1.2.1, 2.1.2.2, 2.2.3.2, 2.2.3.3, 2.2.2.1, 2.2.2.2). The purpose of this choice is twofold: on one hand, we want to determine at which level of accuracy different measures on suffix graphs reconstruct the SCOP classification. For example, if a measure correctly separates domains in different classes but not in different folds, we can conjecture that the measure grasps information encoded in the dominant secondary structures and not in their spatial arrangement. On the other hand, we want to test whether the primary sequence of domains systematically differs from random strings, and if so whether this difference is a widespread phenomenon or it is confined to specific leaves of the classification. To do so, we analyze 100 random permutations of each string in D_1.

As shown in Figure 10, all domains use between 15 and 20 symbols, and have empirical entropy[5] between 3.5 and 4.2; entropy in the same SCOP leaf can vary

[4]For each domain, we use at most three homologues coming from different species. Our choice of branches at each level, and of domains in each class, is arbitrary. For reasons of practical efficiency, domains in D_1 have length between 40 and 200.

[5]By "empirical entropy" we mean the approximation of the entropy of the ergodic source that generated each sequence using the observed frequency of amino acids.

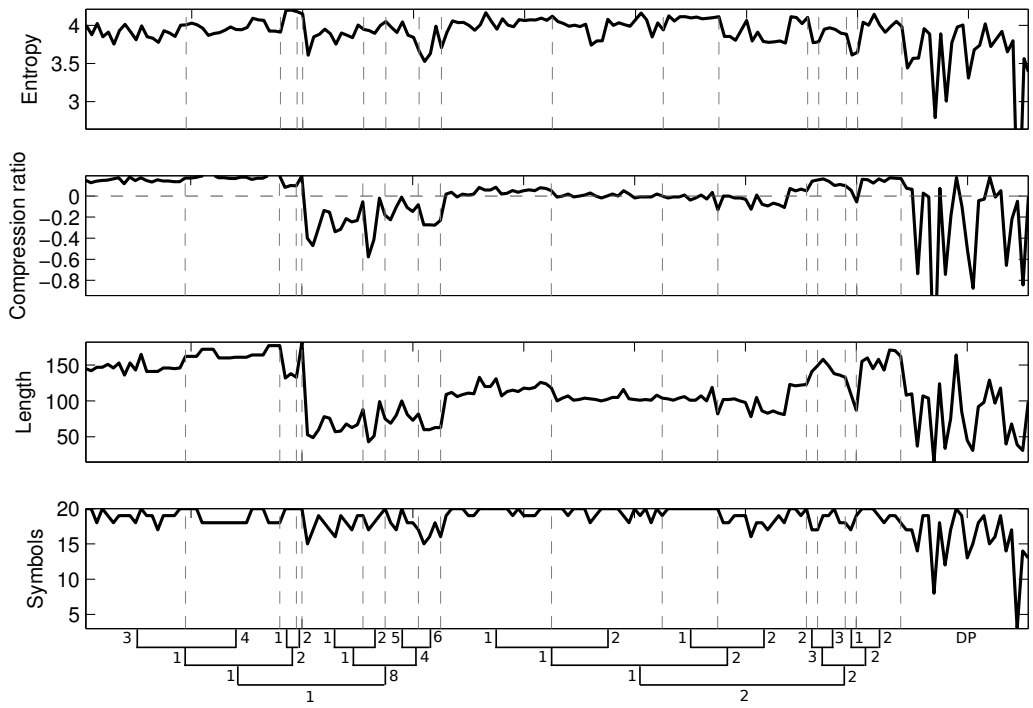

Figure 10: An overview of strings in D_1 and D_2. Empirical entropy is computed using logarithms to base 2 ($0 \cdot \log_2(0)$ is set to 0). Compression ratio is defined as $(|s| - |s'|)/|s|$, where s is the original string, s' is its compression with `gzip -9`, and $|\cdot|$ is file size in bytes.

widely inside this range, however. Conversely, string length and the compression ratio achieved by a popular string compressor, are uniform inside many leaves (e.g. 1.1.1.3, 1.1.1.4, 1.1.2.1, 1.8.4.6, 2.1.1.1, 2.1.1.2, 2.1.2.1, 2.1.2.2, 2.2.3.3, 2.2.2.2), and they display trends that are very similar to each other. Not surprisingly, a significant proportion of domains are either expanded or not compressed at all.

Modern proteins do not consist entirely of domains: some regions have no fixed spatial configuration under physiological conditions, but are capable of dynamically transitioning through an ensemble of structures [136, 154, 180]. The flexibility of these unstructured (or *disordered*) segments allows them to fold and bind to a target simultaneously, transitioning from disorder to order according to their biochemical environment: this allows a single protein to bind multiple targets, and different proteins to bind the same target, an important feature in signalling and regulation

networks. The fluctuation of spatial conformation is also exploited to create regions of exclusion in space, to facilitate phosphorylation and acetylation, and to capture small molecules. In disordered regions, the relationship between sequence and structure is different than in typical folded domains: disordered regions are known to be enriched in charged and polar, and depleted in hydrophobic residues. Along with other chemical and spatial indicators, these biases have been used to construct various disorder prediction heuristics, and to classify disordered regions into subclasses [96]. From the purely syntactic viewpoint, disordered regions tend to have low entropy [146, 172], however some disordered sequences have high entropy, and some low-entropy sequences are not disordered.

DISPROT [154] is a comprehensive functional classification of all polypeptide regions for which there is experimental evidence of disorder. We collect a subset of 23 regions of DISPROT in a secondary dataset (called D_2 in what follows); the choice of strings in this set is again arbitrary, except that, for efficiency and consistency, we consider only proteins with a single disordered region of length at most 200. The purpose of this dataset is twofold: on one hand, we want to test whether disordered regions differ from domains according to suffix graph measures. We conjecture that if a measure clearly separates D_1 from D_2, then it grasps information that only polypeptides with a fixed spatial conformation encode. On the other hand, we want to test whether disordered regions can be distinguished from random strings, and whether such difference resembles those that intercur between domains and random strings. To do so, we analyze again 100 random permutations of each string in D_2.

As shown in Figure 10, just 8 regions in D_2 use less than 15 symbols, and just 5 regions have empirical entropy less than 3.5, the minima in D_1. Furthermore, just 8 regions have positive compression ratio, and compression ratios in D_2 are never larger than the largest compression ratio achieved in D_1. Therefore, strings in D_2 do not appear as systematically "less complex" than strings in D_1. Six disordered regions

have compression ratio smaller than -0.58, the minimum in D_1, but this does not allow to conclude that disordered regions are systematically "more complex" than strings in D_1 either.

1.7 Laws governing polypeptides

In this section we investigate the dependence of suffix graph measures on string length and on the the hiatus of subsequences in datasets D_1 and D_2.

1.7.1 Dependence on string length

Preliminary analyses suggest that the number of special points taken relative to the total number of points is inversely proportional to string length. At $\omega = 1$ this inverse proportionality comes not unexpected: the total number of points (in this case, distinct substrings) grows at most as the square of string length, and the number of special points (in this case equivalent to the number of distinct special substrings) grows at most linearly with string length, therefore the number of special points divided by the total number of points should behave like $a/n + b$, where n is string length and a, b are suitable constants, assuming that the strings in the dataset are approximately random. In principle, every string in $D_1 \cup D_2$ could obey a different set of parameters, making $D_1 \cup D_2$ appear as a disordered cloud in the plane with dimensions (Special points / Total points) and string length. We expect, however, to see a limited number of distinct curves along which domains in similar SCOP groups align. These curves (that we will also call *loci* in what follows) should be signatures of such groups, and their detection could guide classification.

Plotting the relative number of special points versus string length at $\omega = 1$ (Figure 11) shows indeed the expected $1/x$ proportionality but, surprisingly, most strings in $D_1 \cup D_2$ *are aligned along the same locus* with coefficients[6] $a \approx 1.435$, $b \approx 0$.

[6]All the coefficients reported here are computed using the `fit` function of the MATLAB curve fitting toolbox. A detailed investigation of the coefficients of such best interpolations is outside the

Significantly, the only strong outlier is DISPROT 34. There are three features that make DISPROT 34 unique in the dataset: its highly repetitive structure, the use of just 3 distinct symbols, and its small entropy (≈ 1.215). The locus could therefore reflect a property of all strings that lack a strong periodic structure, that have a sufficiently high number of symbols, a sufficiently high entropy, or any combination of these three features. To test this hypothesis, we collect an additional dataset consisting of 89 distinct SCOP domains of length at most 30: we will refer to this set as D_3 in what follows. It turns out that D_3 contains at least two strings[7] that lack a strong periodic structure, use a number of symbols comparable to strings in $D_1 \cup D_2$ (14 and 13), have entropy comparable to strings in $D_1 \cup D_2$ (≈ 3.5398 and ≈ 3.8643), but that do not lie on the locus. This proves that the locus cannot be explained by any combination of the candidate quantities alone. We also note that low entropy alone does not expel a string from the locus: Figure 12 shows that random strings on 20 symbols and minimum entropy (≈ 1.1169, even lower than DISPROT 34) can lie on the curve.

A locus with similar parameters persists at $\omega = 2, 3$, with DISPROT 34 and some strings in D_3 continuing to be outliers. Increasing ω beyond 3 gradually transforms the sharp locus into a dispersed cloud that keeps no resemblance to the original curve. At $\omega \geq 6$ three disordered regions (DISPROT 34, 13 and 19) become clearly separated from the rest of the dataset, along with few strings in D_1.

We expect a direct proportionality between the number of special points y (not normalized) and string length n, and in particular a linear relationship $y = an + b$ when $\omega = 1$. Plotting these two quantities together shows indeed a linear bundle centered around $a \approx 0.317$, $b \approx 0.318$ for all domains and disordered regions except

scope of this chapter.

[7] Tumor necrosis factor receptor superfamily member 17, BCMA; and Nucleic acid binding protein p14.

DISPROT 34 and 25[8] (Figure 13). The locus remains linear at $\omega = 2$, but from $\omega = 3$ it progressively tends towards a sparse nonlinear shape that we will call *horn* in what follows. We note that, at $\omega \geq 6$, this shape includes strings that were outliers at lower ω, in particular the highly regular DISPROT 34. Strings in D_3 are never seen to escape the locus, but random strings on 20 symbols with minimum entropy turn out to be outliers for every ω, proving that the curve is not an unavoidable regularity of all strings.

The total number of points assumes the expected quadratic shape at $\omega = 1$, which persists up to $\omega = 4$, then it gradually becomes a horn (Figure 14). Neither D_3 nor DISPROT 34 escape the locus, but few other disordered regions do. As before, it can be shown that there is at least one string that does not lie on the locus.

Similar curves appear when other measures are considered: in all cases, at most one sharp curve appears, collecting most of $D_1 \cup D_2$ with the possible exception of few outliers. Rather than analyzing each one of these curves in detail, we prefer to focus on two measures in which the shape of the locus changes in a different way as a function of ω. In the relative number of antispecial points (Figure 15) a sharp nonlinear curve of direct proportionality with few, strong outliers persists up to $\omega = 2$, then it becomes disordered at $\omega = 3, 4, 5$, and finally it transitions towards a bundle of inverse proportionality at $\omega = 8$. The second notable example is the relative number of normal points: no clear locus appears at $\omega \leq 6$, but a linear bundle starts to emerge at $\omega = 7, 8$.

Measures on *subsequences*, on the other hand, depend weakly on string length. For example, the number of antispecial subsequences (not normalized) grows exponentially with string length, and their growth is confined inside a bundle whose width expands with length (Figure 16). However, there is no correlation between the *relative* number of antispecial subsequences and string length when $\omega > 1$ (Figure 17).

[8]Part of Fibronectin-binding protein A.

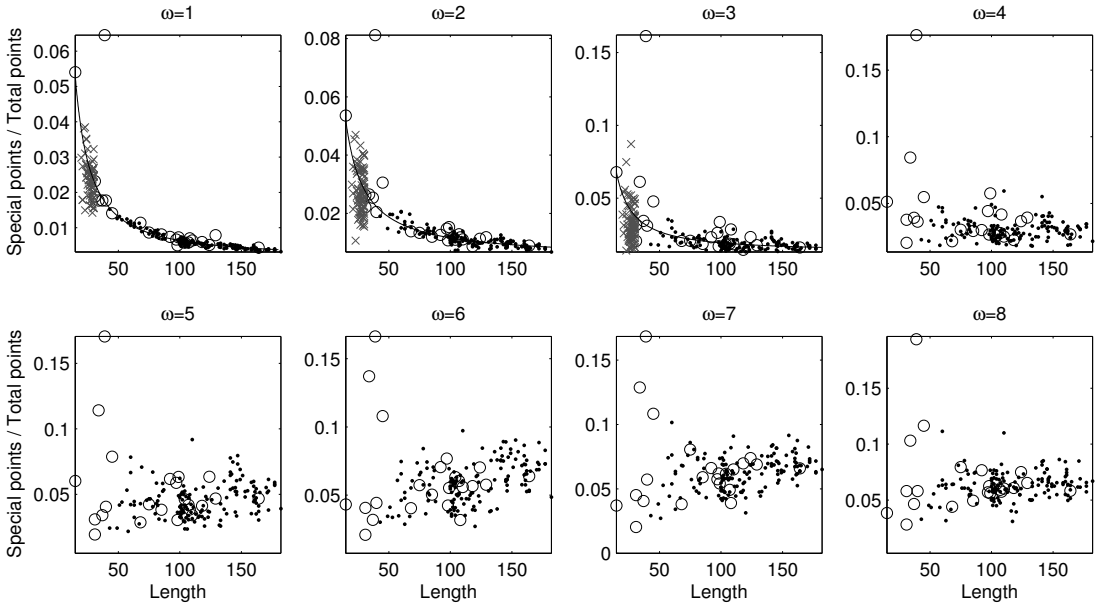

Figure 11: Relative number of special points versus string length, in domains (dots) and disordered regions (circles). Strings in D_3 are represented as gray crosses. The best interpolating $a/n + b$ curve is shown as a black line.

On the other hand, the relative number of normal subsequences depends on string length under a relationship of exponential inverse proportionality for all $\omega \leq 5$; this locus disappears into a disordered cloud at $\omega \geq 6$.

A measure can have no dependence on string length at a specific ω, but it could nonetheless obey other rules. We have just seen that the relative number of normal points has no locus at $\omega < 7$, and that the relative number of normal subsequences has no locus at $\omega \geq 6$: plotting one of these two measures versus the other shows again no correlation for $D_1 \cup D_2$, but it reveals that D_3 is aligned along a horn at all values of $\omega > 1$ (Figure 18). No such regularity occurs, however, when we plot the relative number of antispecial points versus the relative number of antispecial subsequences.

1.7.2 Dependence on ω

In the previous section we have seen that the shape of the curves relating suffix graph measures to string length changes with ω: thus, it is natural to investigate the

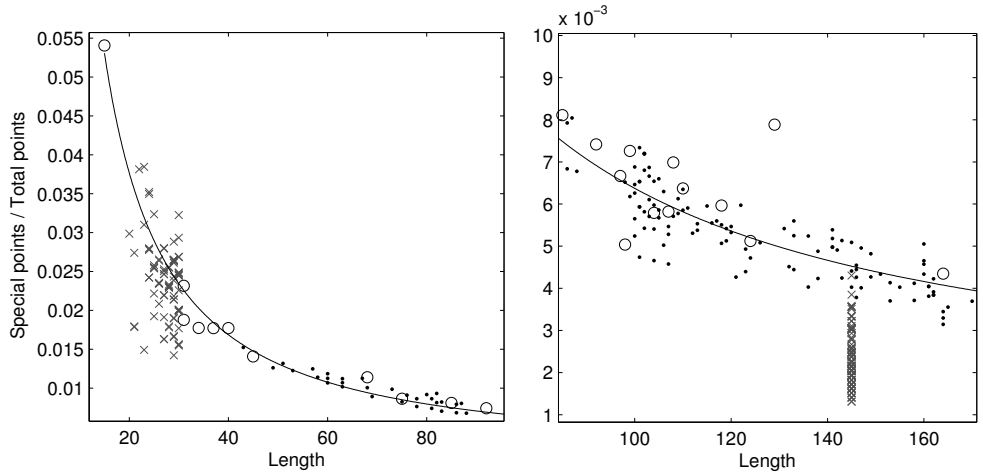

Figure 12: The graph of Figure 11 at $\omega = 1$. (Left) Some strings in D_3 (gray crosses) do not lie in the locus. (Right) Some random strings on 20 symbols and minimum entropy (gray crosses) enter the locus.

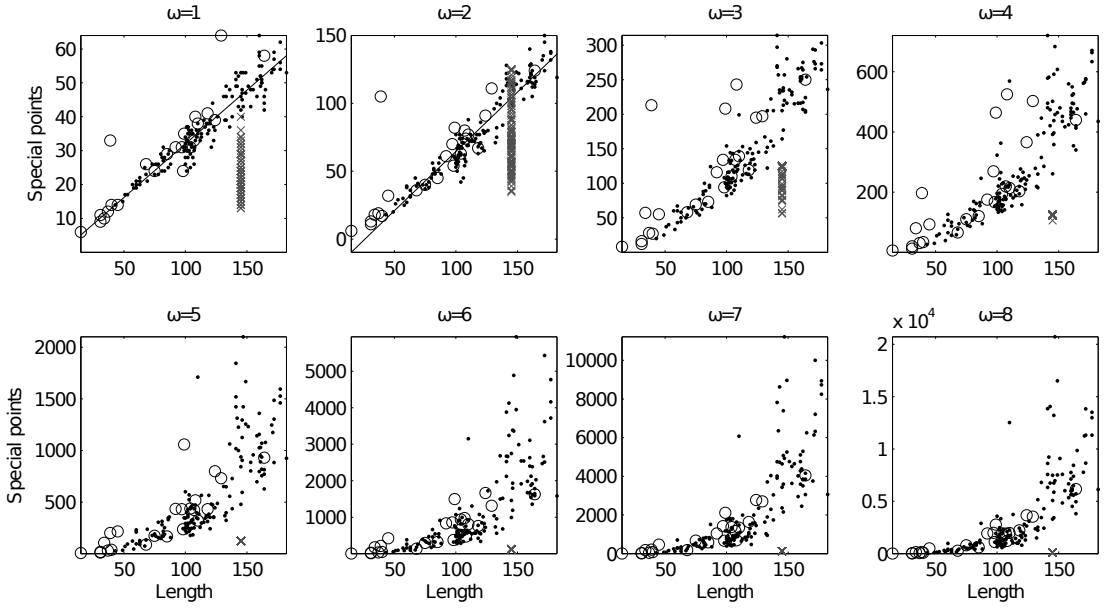

Figure 13: Number of special points (not normalized) versus string length, in domains (dots) and disordered regions (circles). Gray crosses are random strings on 20 symbols with minimum entropy.

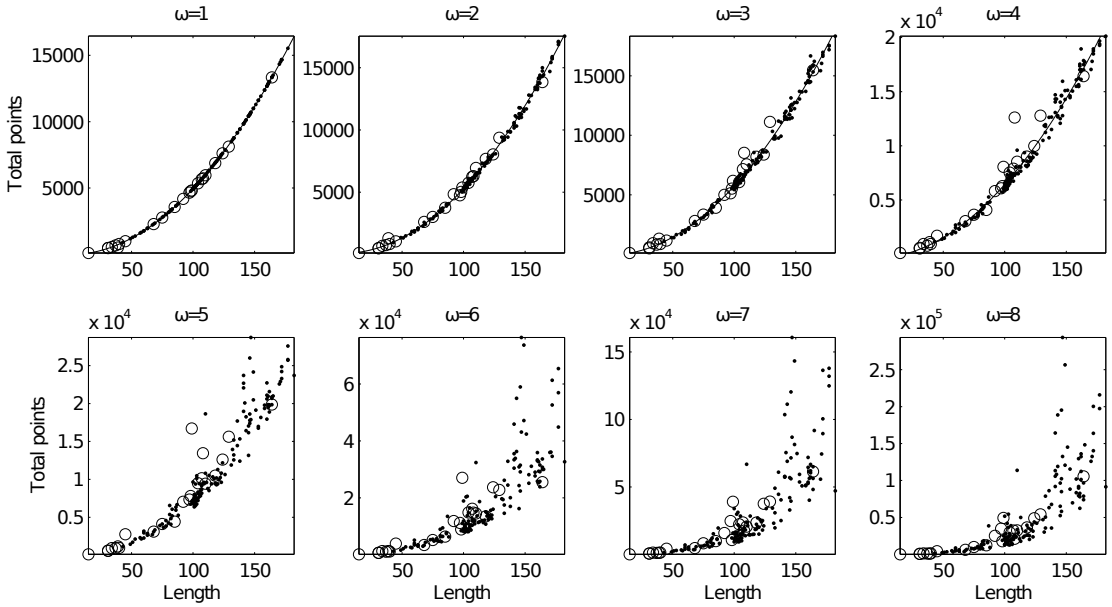

Figure 14: Total number of points versus string length, in domains (dots) and disordered regions (circles). The interpolating line is $y = a \cdot n^2 + b \cdot n + c$. At $\omega = 1$, $a \approx 0.4963$. At $\omega = 2$, $a \approx 0.5453$. At $\omega = 3$, $a \approx 0.5585$. At $\omega = 4$, $a \approx 0.6057$.

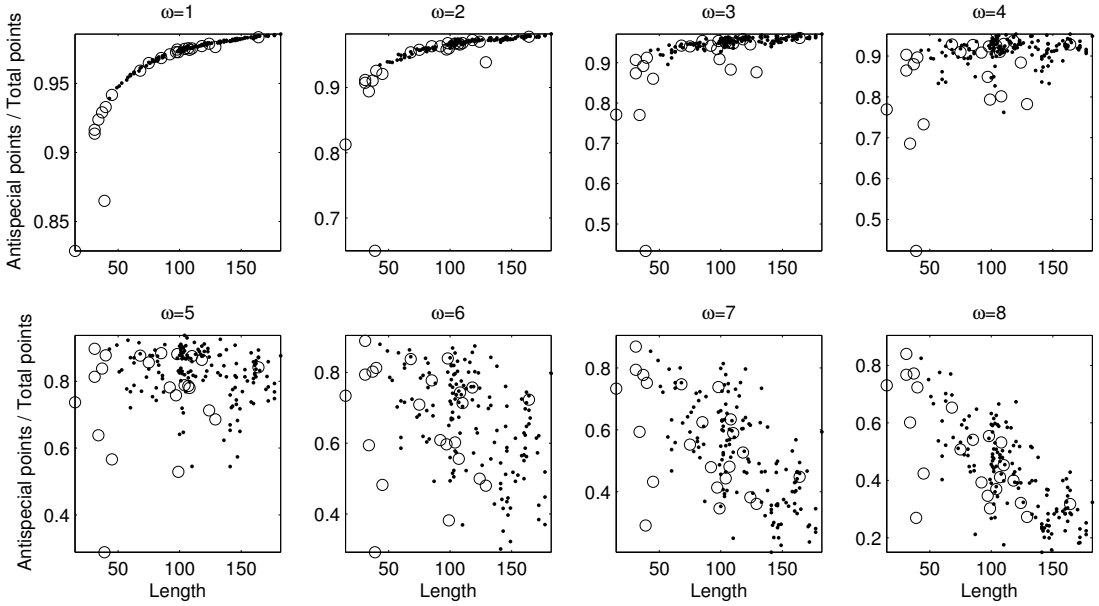

Figure 15: Relative number of antispecial points versus string length, in domains (dots) and disordered regions (circles). At $\omega = 2, 3, 4$ some outliers fall outside the displayed range.

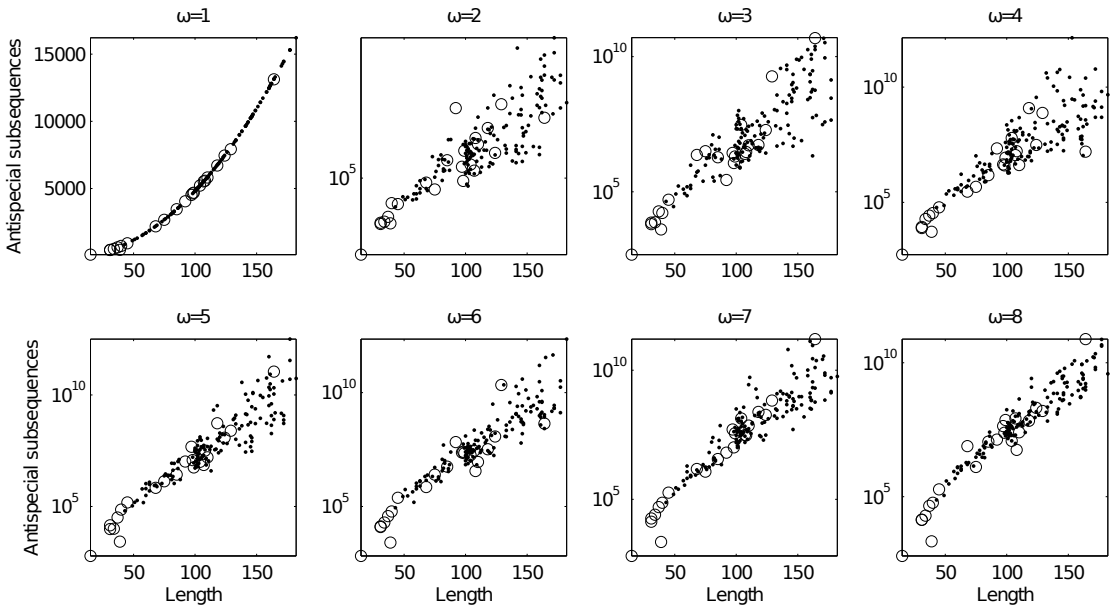

Figure 16: Number of antispecial subsequences versus string length, in domains (dots) and disordered regions (circles).

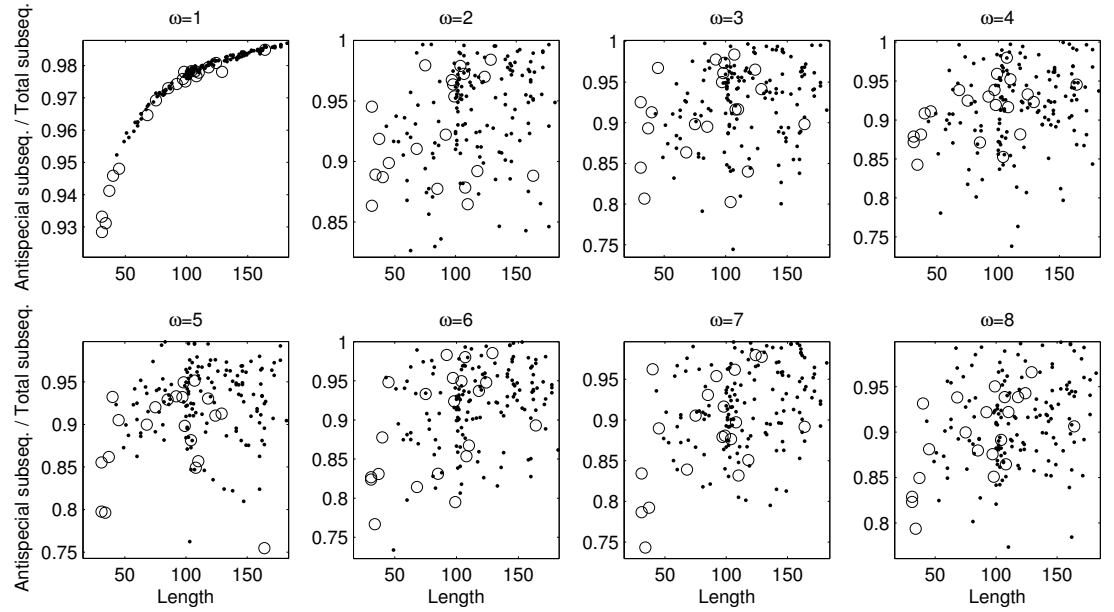

Figure 17: Relative number of antispecial subsequences versus string length, in domains (dots) and disordered regions (circles). At $\omega > 1$ few strings lie far from the main cloud, and fall outside the displayed range.

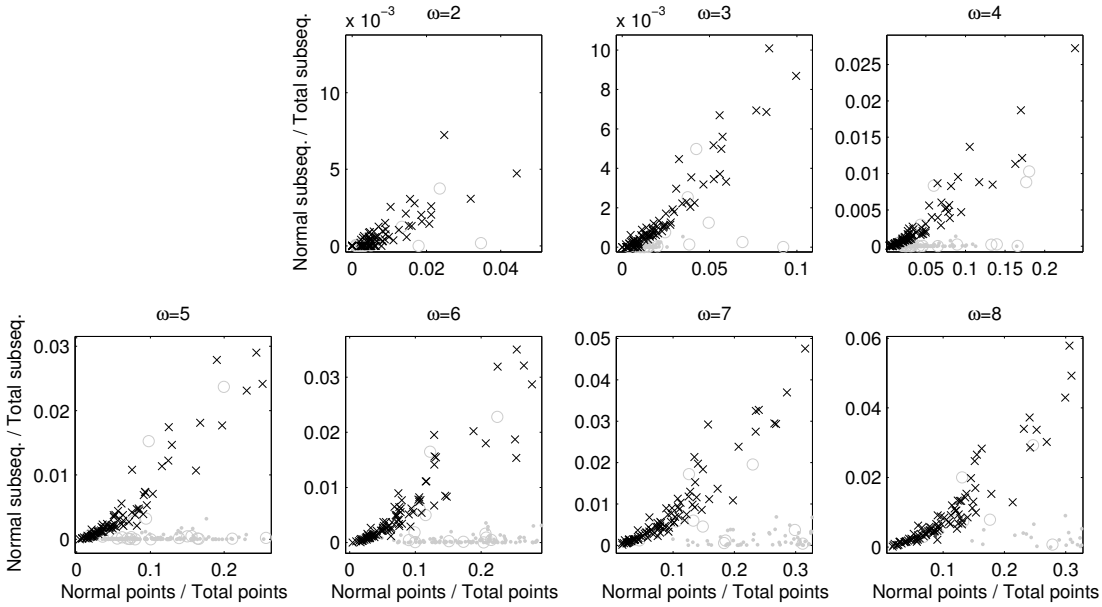

Figure 18: Relative number of normal subsequences versus relative number of normal points, in D_3 (black crosses), D_1 (light gray dots) and D_2 (light gray circles).

dependence of these measures on ω itself. Plotting the relative number of special, antispecial, normal and terminal points on the same graph reveals a recurrent motif: *many strings pass through five phases*, marked by the following events (Figure 19): (1) the increase of special points above terminal points; (2) the increase of normal points above terminal points; (3) the overtaking of special points by normal points; (4) the final overtaking of antispecial points by normal points, after which normal points become the most abundant category in suffix graphs.

Studying the whole datasets reveals that the values of ω at which each of these transitions occurs is not constant across D_1 (Figure 20): while there is little variation for the value at which normal points overtake terminal and special points (always around 4,5), and at which special points overtake terminal points (always around 3,4), the value at which normal points overtake antispecial points varies significantly, and it does not respect SCOP boundaries. Superfamily 1.1.1 and fold 2.2 transition at $\omega \geq 6$, while fold 1.8 either presents no phase transition or transitions at $\omega = 8$. In family 1.8.1.1 normal points never overcome antispecial points, and special and

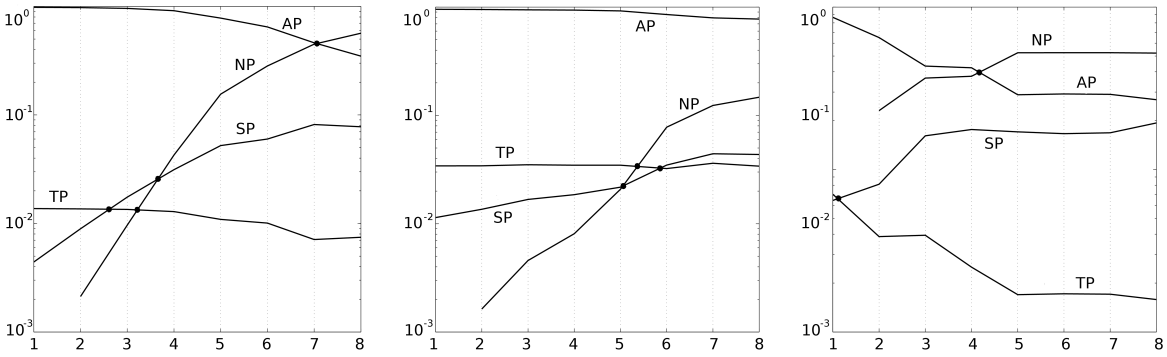

Figure 19: Dependence of the relative number of special, antispecial, normal and terminal points on ω. (Left) A member of family 1.1.1.1. (Center) A member of family 1.8.1.1. (Right) DisProt 34.

normal points increase above terminal points at relatively high ω. Other exceptions to the motif described above occur in D_2, where, not surprisingly, DISPROT 34 is an exemplar anomaly (Figure 19).

This extended heterogeneity prompts us to test whether the relationship between each suffix graph measure and ω is controlled by general laws, like those seen for string length (Figure 21). It turns out that the relative number of special, antispecial, normal and terminal points trace wide sigmoid bundles, in which all strings have similar shape but possibly very different values[9]. These loci are not universal: apart from few clear outliers in D_2 (DISPROT 34 in special and normal points, DISPROT 13 in special points, DISPROT 9 in antispecial and terminal points, DISPROT 20 in normal and terminal points), strings in D_3 turn out to follow very different rules (Figure 21). Similar conclusions can be drawn for the absolute number of points and subsequences.

1.8 Laws governing random permutations of polypeptides

The alignment of most polypeptides along the same loci, and the presence of outliers to such loci, proves that the strings in $D_1 \cup D_2$ follow a specific compositional pattern,

[9]There is a tendency for strings in folds 1.1-2.2 to assume a smaller proportion of antispecial and terminal points, and a larger proportion of special and normal points, compared to the rest of D_1. We leave this detail to future research.

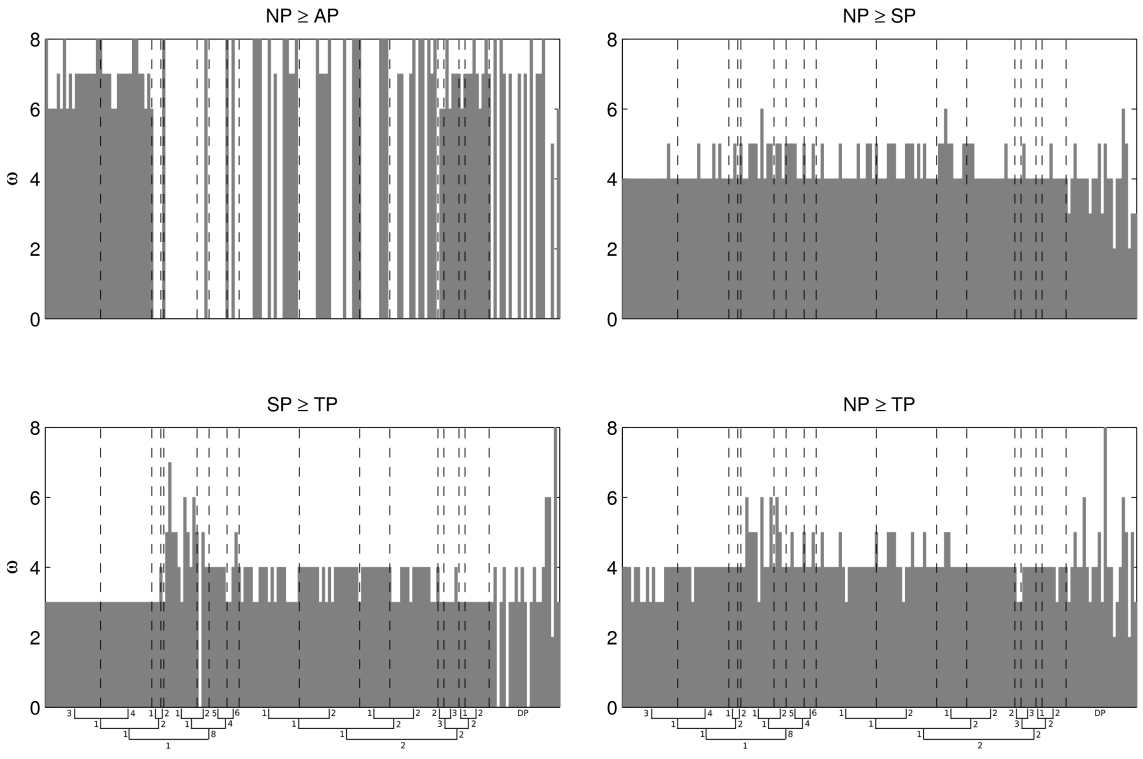

Figure 20: Values of ω at which phase transitions occur in $D_1 \cup D_2$.

and that this pattern is shared by proteins lying in very different groups of the SCOP hierarchy. We naturally expect the information that encodes these loci to be carried by the *sequences* of amino acids: if this is the case, randomly permuting the strings would destroy the signal and it would cause the loci to degenerate into random clouds. We analyze a set of 100 random permutations for each string in $D_1 \cup D_2$. Surprisingly, such permutations still trace the same loci in all graphs of the previous sections (see e.g. Figure 22): this proves that, in the dataset analyzed, *it is the composition of symbols, not the sequence, to be responsible for the alignment of polypeptides along regular loci*. However, the relationship between distribution of symbols and locus is not bijective: among the members of D_1 there are significant variations in how the frequencies of symbols are distributed, therefore a locus does not necessarily imply similar distributions of symbols.

On the other hand, sequence does influence suffix graph measures: in Figures 12

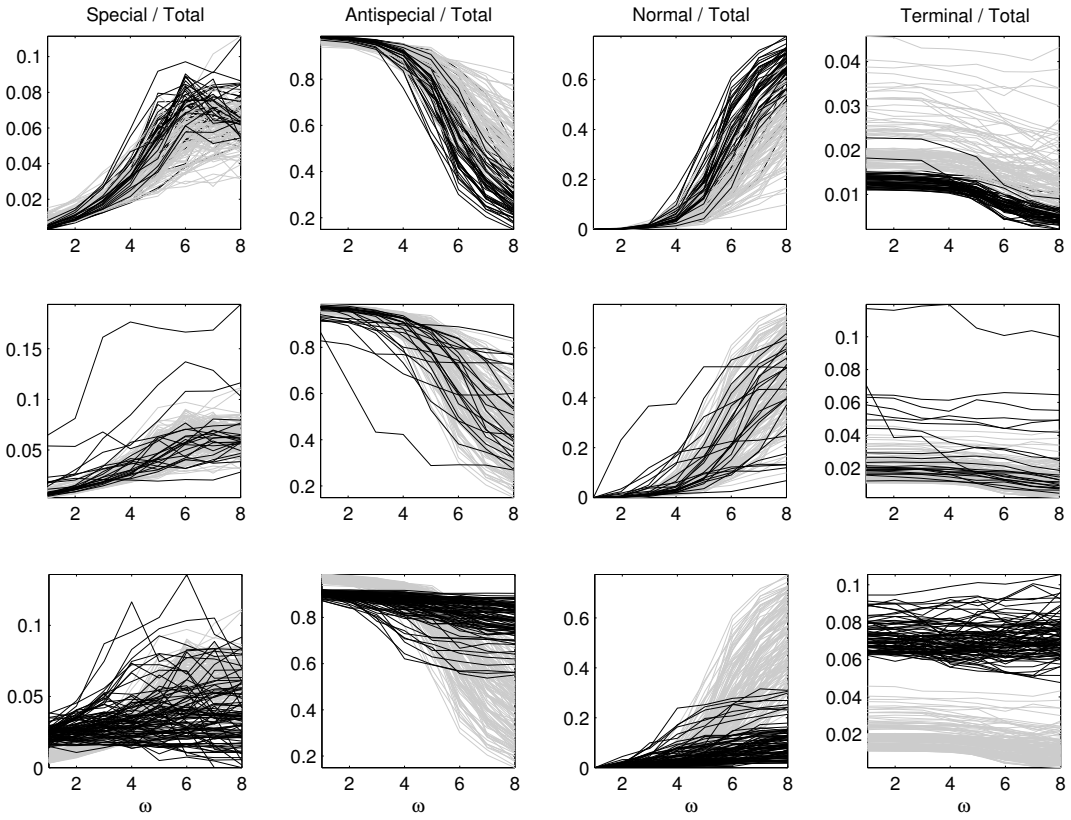

Figure 21: Relative number of points versus ω. (First row) D_1: folds 1.1-2.2 (black) compared to 1.8-2.1 (gray). (Second row) D_2 (black) compared to D_1 (gray). (Third row) D_3 (black) compared to D_1 (gray).

and 13, for example, random arrangements of a set of 20 symbols with minimum entropy trace a range of different values. Thus, in the strings of $D_1 \cup D_2$, *the effect of sequence on suffix graph measures is weaker than the effect of symbol composition.*

The influence of sequence on suffix graph measures deserves more attention. We project the 100 random permutations of each string in $D_1 \cup D_2$ onto the space generated by every possible (x, y) pair, where x and y are suffix graph measures. Let's concentrate on the relationship between special points and total points first: the visual inspection of the graphs of few strings in D_1 shows that *the set of random permutations forms a well defined, linear bundle* at $\omega = 1$ and $\omega \geq 6$, while random clouds appear at $2 \leq \omega \leq 5$ (Figure 23). Polypeptides are always seen to belong to these bundles.

Probing the extent to which this relationship is supported by D_1, D_2 and D_3 is clearly unfeasible by drawing and analyzing each graph visually. Therefore, we measure the correlation coefficient between special and total points for each string[10]. The graph of the correlation coefficients for all three datasets shows that at $\omega = 1$ the random permutations of all strings have a strong linear negative correlation, except for DISPROT 34 and few members of D_1 and D_3 (Figure 24). At $\omega = 2, 3, 4$ correlation progressively becomes weaker, except in DISPROT 34 in which it is strong and positive for all $\omega > 1$ (Figure 25). At $\omega \geq 5$ correlation progressively becomes strong and positive for most strings in $D_1 \cup D_2$, but it remains weak in most of D_3 (except, e.g., Thyroid receptor interacting protein 6, *Homo sapiens*), in some members of D_2 (e.g. DISPROT 20), and in fold 1.8 (e.g. in the C-terminal domain of γ,δ resolvase, *Escherichia coli*). In all cases in which most strings have a strong correlation, strings with a weak correlation can be found, and vice versa, proving that the pattern of strong and weak correlations as a function of ω is not an unavoidable regularity of all

[10]The correlation coefficient represents the strength of linear relationship between two variables as a value between -1 (strong linear negative relationship) and 1 (strong linear positive relationship). We consider "strong" a correlation that has absolute value ≥ 0.5 and p-value $\leq 10^{-5}$.

polypeptides.

The pattern of correlations neither reveals clear distinctions among SCOP groups, nor between polypeptides and their permutations: all strings in the datasets belong to the linear loci of their random permutations, with the exception of DISPROT 28, 25 and 34 at $\omega = 1$ (Figures 25 and 26). The coefficient a of the linear interpolations is approximately constant inside $D_1 \cup D_2 \cup D_3$ at $\omega = 1$, and approximately constant in $D_1 \cup D_2$ at $\omega \geq 6$. The coefficient b, on the other hand, oscillates widely across $D_1 \cup D_2$.

Analyzing in detail the effect of sequence on each suffix graph measure is outside the scope of this section. Here we just observe that, except for few cases, normal points are not strongly correlated to total points at $\omega \leq 4$, but at $\omega \geq 5$ $D_1 \cup D_2$ reaches a strong correlation, while the correlation of D_3 remains lower (Figure 27). Antispecial points are highly correlated with total points at all values of ω; terminal points are not correlated to total points at $\omega = 1$, but they become strongly correlated at $\omega = 2, 3$, and finally their correlation stabilizes around a lower value at $\omega \geq 5$.

1.9 Conclusion and extensions

Some natural measures on the abundance of points, arcs and subsequences in suffix graphs reveal a high degree of structure in ordered, artificial strings, and detect a specific, previously unknown set of rules related to string length and to the hiatus of subsequences in a range of structurally and functionally diverse polypeptides. Conforming with the current consensus that sees proteins as random strings, these rules are influenced by the distributions of symbols more strongly than by their organization within the sequence. In most polypeptides, it is seen that even their random permutations amass along specific linear loci. Counterexamples show that none of such rules is an unavoidable property of all polypeptides or of all distributions of symbols, thereby suggesting that the shapes of these loci are specific signatures of the

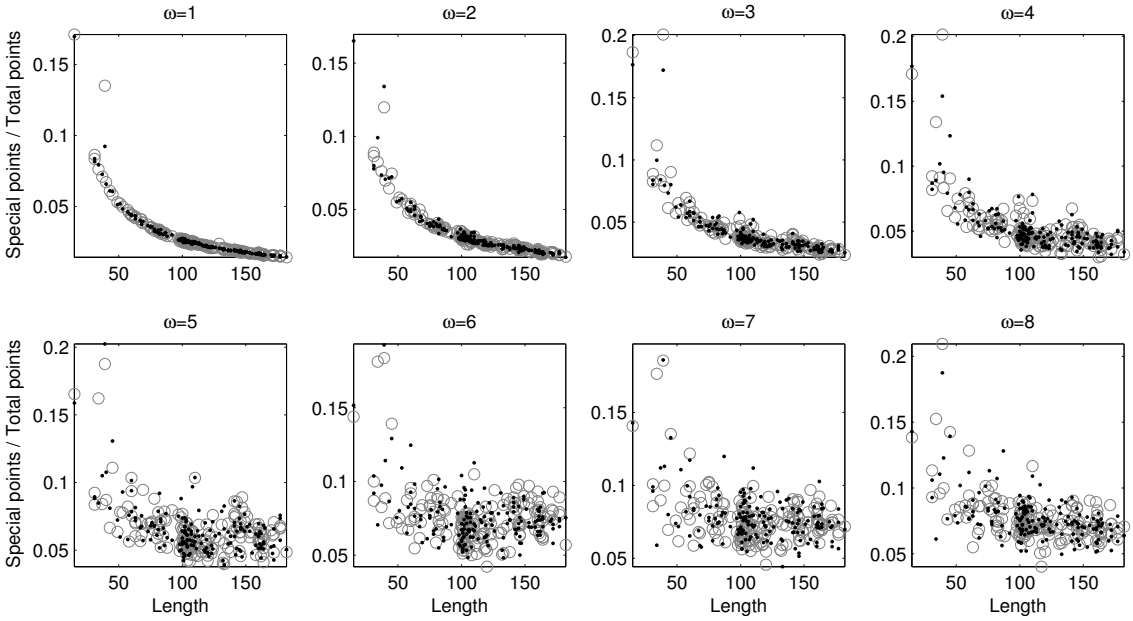

Figure 22: Relative number of special points versus string length in $D_1 \cup D_2$ (light gray circles), and in a copy of $D_1 \cup D_2$ in which each string has been randomly permuted (black dots).

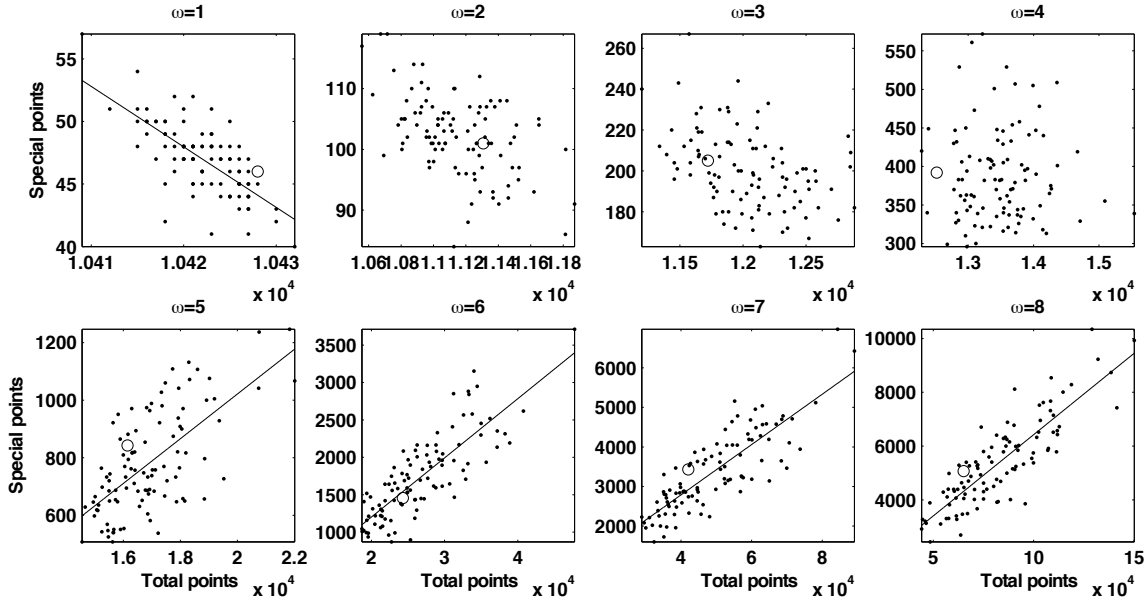

Figure 23: Number of special points versus total number of points in Hemoglobin I from *Scapharca inaequivalvis* (a protein in D_1, circle) and in 100 random permutations of its sequence (dots). Lines indicate the best linear interpolation of the set of permutations. Other strings in D_1 trace similar shapes.

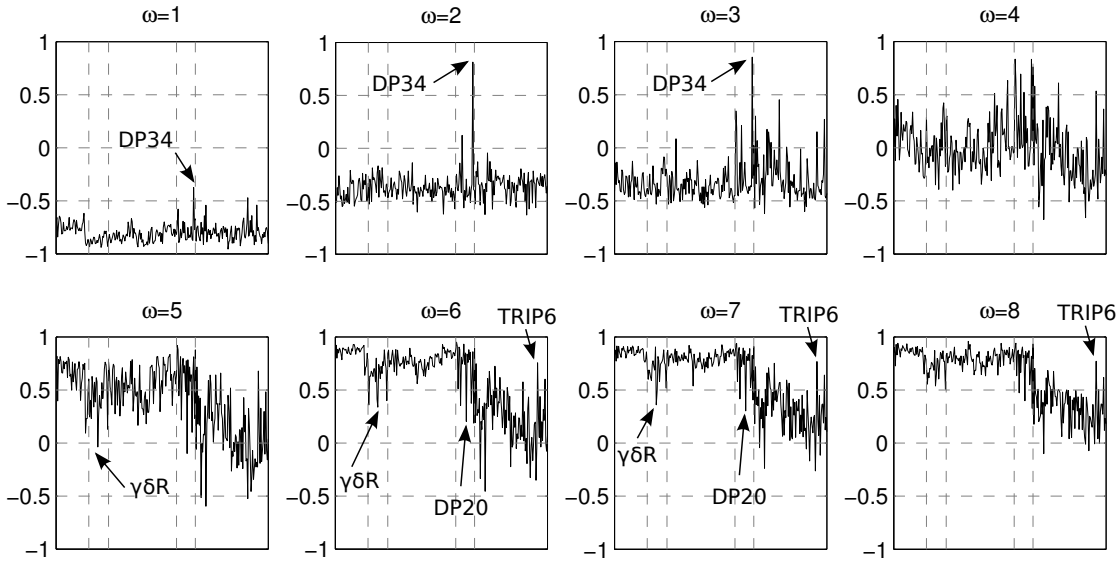

Figure 24: Correlation between the number of special and total points in the dataset. For clarity of presentation, p-values are not shown: they are $\leq 10^{-5}$ wherever the correlation is ≥ 0.5 in absolute value. Vertical dashed lines highlight fold 1.8 and D_2. $\gamma\delta R$: C-terminal domain of γ,δ resolvase, *Escherichia coli*. TRIP6: Thyroid receptor interacting protein 6, *Homo sapiens*.

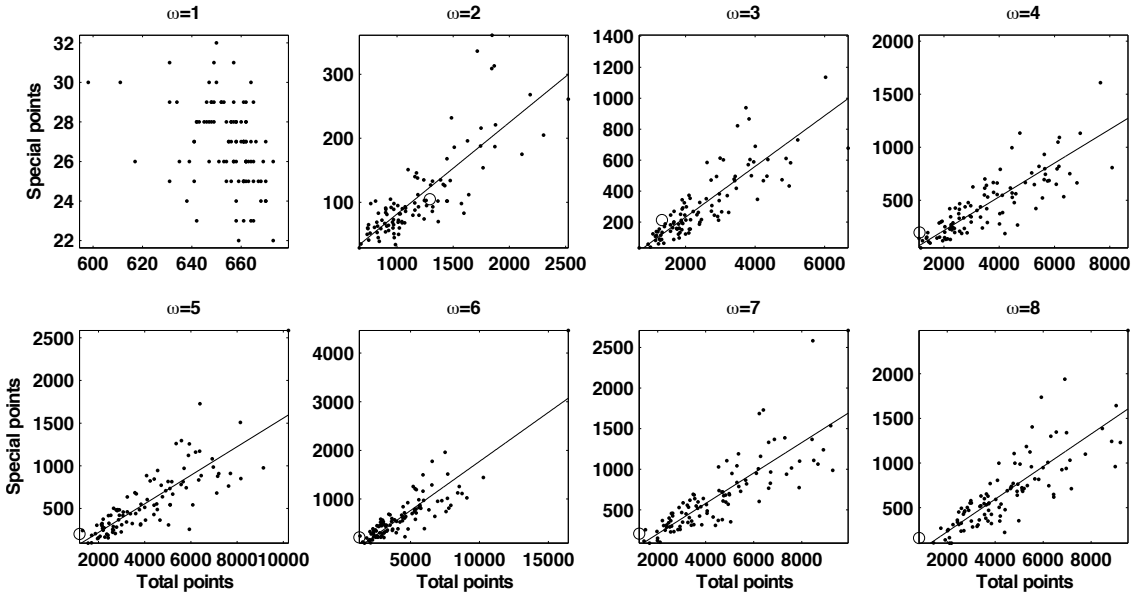

Figure 25: Number of special points versus number of total points in DISPROT 34.

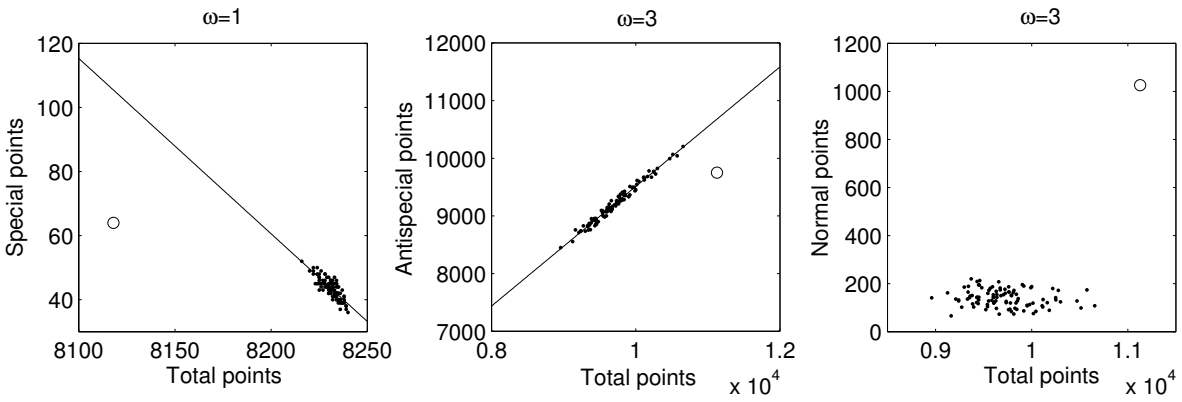

Figure 26: Number of special (left), antispecial (center), normal (right) points versus number of total points in DISPROT 25.

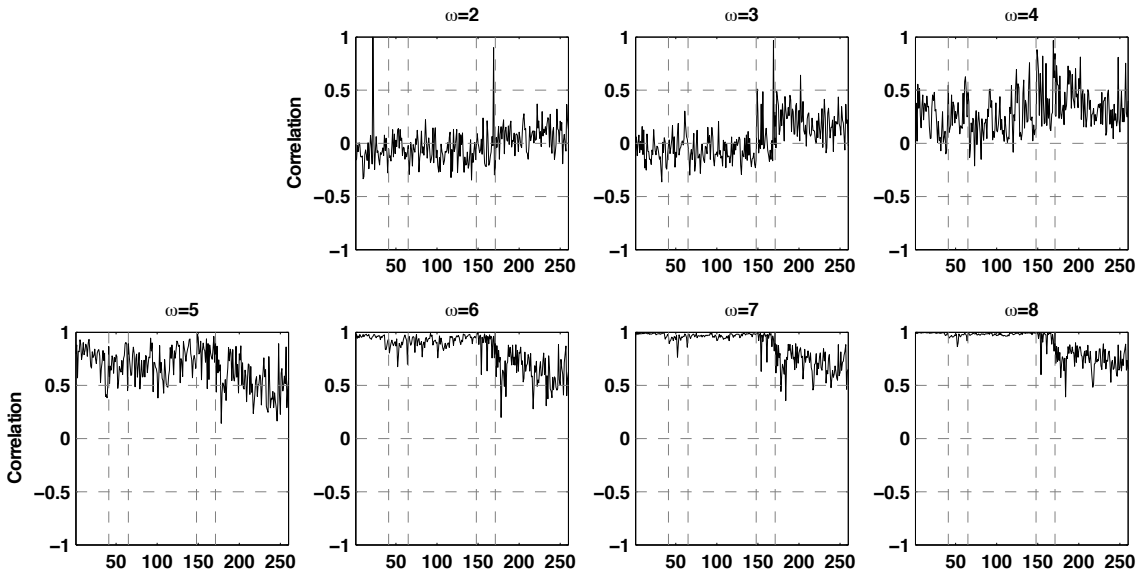

Figure 27: Correlation between the number of normal and total points in the dataset. Vertical dashed lines highlight fold 1.8 and D_2.

dataset under observation or of its parts.

It is well known that amino acid composition varies with functional class and cellular localization [82]. The fact that most of the rules described in this paper hold for structurally and evolutionarily diverse polypeptides might suggest that they capture organizational constraints that cross the boundaries of protein families, and which could be implied in the chemical or spatial stability of amino acid chains [71, 72], or in the mechanism by which secondary structures aggregate and connect to each other [130]. Alternatively, these rules might capture *evolutionary* regularities, for example properties of the limited number of peptides that arguably composed the primitive peptide world [132, 177], or laws behind the assemblage of these original segments into modern domains [106].

It is also well known that amino acid abundance is highly influenced by genome structure, and that it varies with species [82]. The regularities described in this chapter could therefore reflect biases and optimizations in the translation machinery [49, 147, 151, 171, 176], or be the image of corresponding constraints in genomes.

From the experimental viewpoint, this work stimulates the analysis of the whole SCOP with the purpose of counting and mapping the repertoire of rules that occur therein. Studying what happens at higher values of ω would also be a natural extension. From the theoretical viewpoint, this work opens the problem of explaining the effect of the sequence and of the distribution of symbols of a string on the described loci. A related avenue consists in defining a complexity measure for strings and distributions hinged on these rules, and in comparing it to other state-vector complexity measures, like Shannon's entropy and the global compositional measure in [171]. The problems of computing such measures efficiently and of structurally comparing the suffix graphs of different strings lend themselves to the formulation of interesting algorithmic extensions.

CHAPTER II

PHYLOGENY CONSTRUCTION WITH GAPPED PATTERNS

Measures of sequence similarity based on some underlying notion of relative compressibility are becoming increasingly of interest in connection with massive tasks of text classification such as, notably, in document classification and molecular taxonomy on a genomic scale. Sequences that are similar can be expected to share a large number of common *substrings*, whence some successful measures in this class have been based on the substring composition of the input sequences. A family of methods avoids to consider the potentially quadratic number of all distinct substrings by bounding their maximum length and observing convergence when length 5 or larger is reached (see, e.g., [14, 132, 156] for a small sampler, and [169, 170] for a more comprehensive review). A recent alternative points to the linear set of *maximal* substrings – i.e. strings that cannot be extended in any direction without reducing their number of occurrences – as sufficient to grasp essential phylogenetic information (see [7] and references therein). Yet other methods explicitly resort to data compression techniques such as LZ77. Strings with *gaps* or *don't care characters*, on the other hand, have traditionally been used as signatures of protein families or as features in SVM text classification, but they have rarely been applied to phylogeny reconstruction. In particular, no existing application allows a very large proportion of don't care characters in substrings, and no existing application compares the gapped substring composition of two strings.

This chapter explores the potential of carefully chosen gapped substrings to guide

phylogeny reconstruction. First, we measure for the first time the quality of phylogenies constructed by explicitly comparing the *composition* of structures that do allow gaps. For "composition" of a string s we mean here the set of *all* structures of a given type that occur in s. We turn to rigid gapped *motifs* in particular, and we compare their phylogenies both to a reference taxonomy and to those generated by popular string-based alignment-free methods. Second, we study the relationship between classification power and number of gaps. For "classification power" of a given set of motifs we mean here the distance between phylogenies reconstructed from the composition of such motifs and corresponding reference phylogenies. We are specifically interested in testing whether *extremely sparse* motifs carry any phylogenetic signal. To accomplish this goal, we use motifs whose length and sparsity have never been considered before. Even worse than substrings, the number of rigid gapped motifs can grow exponentially in the length of a string. Our third objective is measuring the footprints on classification quality of systematic ways to limit this explosion. We experiment with global and local bounds on the density of motifs, with motifs with high z-score, as well as with *maximal* motifs, i.e. motifs that cannot be made more specific without losing support. Unfortunately, even maximal motifs grow too fast to be practical: we thus consider bases that are capable of generating the whole set of maximal motifs but grow quadratically or linearly in the length of the string.

Scaling to entire genomes and proteomes is a fundamental requisite for alignment-free sequence comparison algorithms. At the genome scale, using a smaller set of features during comparison has a significant impact on storage space and on classification time, besides having inherent information-theoretic implications. As the number of sequenced genomes increases, speeding up feature extraction becomes another key priority. We thus study the robustness of measures of similarity built around the dictionary of LZW – the variant of the LZ78 compression algorithm proposed by Welch – as well as of some of its lossy variants, in phylogeny reconstruction. LZW has both a

practically faster implementation than LZ77, and a significantly smaller vocabulary. We focus in particular on recently introduced *gapped* variants of LZW that are equally straightforward to implement, but allow for a controlled number of don't cares to be injected in the substrings that compose the dictionary. Such gaps allow to apply LZW to domains that tolerate some loss of information in exchange for increased compression. An ingenious disambiguation scheme that resolves gaps in blocks rather than one by one affords compression even in lossless mode.

This chapter is organized as follows: Section 2.1 overviews the state of the art in alignment-free sequence comparison, describes the few existing methods that construct phylogenies from gapped patterns, and traces their roots to early large-scale compositional analyses. Section 2.2 studies the dependence between the quality of reconstructed phylogenies and the density, number of solid characters, and statistical significance of gapped motifs and of some of their compact generators. Experiments with more than 3600 trees built on approximately 4.4 billion motifs show that the average performance of suitably defined sets of gapped motifs is comparable to that of popular string-based alignment-free methods. In particular, *extremely long and sparse motifs* produce phylogenies of the same or better quality than those produced by short and dense motifs. Finally, Section 2.3 experiments with the gapped words in the vocabulary of recently introduced variants of the LZW compression algorithm. Dissimilarity measures based on maximal strings in the dictionary of LZW yield phylogenies that are comparable to state-of-the-art methods on test proteomes. Introducing a controlled proportion of gaps does not degrade classification, and allows to discard up to 20% of each input string.

2.1 State of the art

2.1.1 Alignment-free sequence comparison

Classical techniques for the comparison and classification of sequences do not work in the emerging context of massive data classification: edit distances, for example, become both computationally unbearable and semantically ambiguous when applied to whole genomes. The classical approaches are thus currently being supplanted by global, parameter-free similarity measures (see, e.g., [14, 25, 50, 55, 73, 77, 93, 119, 132, 168, 167, 170, 181]), whose theoretical substrates try in various ways to approximate universal measures of mutual information. In their practical implementations, most of these global approaches resort, implicitly or explicitly, to the substring composition of the input.

Substring composition is explicitly used in measures based on the frequency of k-mers, pioneered by Blaisdell [25] and currently being applied to an increasing number of proteomes (see e.g. [73, 132]). Let Σ be a finite alphabet. The k-mer approach projects a string $x \in \Sigma^+$ onto a *composition vector* $\mathbf{X} \in \Re^{|\Sigma|^k}$, with one component for each string $w \in \Sigma^k$. Let $f(w)$ be the observed number of (possibly overlapping) occurrences of w in x, and let $\bar{p}(w)$ be the probability of seeing string w in x according to a $(k-2)$th order Markovian random source. Then, the value $\mathbf{X}[w]$ of $\mathbf{X}$ along component w is a function of the empirical probability $p(w) = f(w)/(|x| - |w| + 1)$ of observing string w in x, purified from noise as follows [132]: $\mathbf{X}(w) = (p(w) - \bar{p}(w))/\bar{p}(w)$ if $\bar{p}(w) \neq 0$; $\mathbf{X}(w) = 0$ otherwise. When the composition vector $\mathbf{X}$ is regarded as a point in a Euclidean space, the similarity between two vectors can be measured by the cosine of the angle between them: this yields the metric $d_{cos}(x, y) = (1 - \cos(\mathbf{X}, \mathbf{Y}))/2 \in [0, 1]$. When $\mathbf{X}$ and $\mathbf{Y}$ are regarded as probability distributions p_x, p_y over Σ^k (cf., e.g., [156]), one may use instead classical information-theoretic measures of dissimilarity, such as the Jensen-Shannon divergence $d_{js}(p_x, p_y) = (KL(p_x|q) + KL(p_y|q))/2$, where $q = (p_x + p_y)/2$, and $KL(p_x|p_y)$ is

the Kullback-Leibler divergence between distributions p_x and p_y, that is, the expected number of extra bits needed to identify a value $w \in \Sigma^k$ drawn from p_x, if a code is used corresponding to the probability distribution p_y, rather than p_x.

A related measure of dissimilarity between probability distributions on Σ is the expected length of the description of a value $a \in \Sigma$ drawn from distribution p, using a code tailored to q. If p and q are IID, this amounts to:

$$\tilde{d}(p|q) = -\sum_{a \in \Sigma} p(a) \log q(a) = -\mathbb{E}_p(\log q(\Sigma))$$

If p and q are Markovian, this can be expressed as:

$$\tilde{d}(p|q) = \lim_{k \to \infty} -\frac{1}{k} \sum_{w \in \Sigma^k} p(w) \log q(w) = -\mathbb{E}_p(\log q(\Sigma))$$

Like KL divergence, $\tilde{d}(p|q)$ violates symmetry and triangle inequality, but it has the nice property that, if x and y are strings generated by Markovian distributions p and q, then $\tilde{d}(p|q)$ is approached by the estimate $d_\ell(x, y) = (d'_\ell(x, y) + d'_\ell(y, x))/2$ as $|x|, |y| \to \infty$ [167], where

$$d'_\ell(x, y) = \frac{\log(|y|)}{\sum_{i=0}^{|x|-1} \ell_{x,y}(i)/|x|} - \frac{\log(|x|)}{\sum_{i=0}^{|x|-1} \ell_{x,x}(i)/|x|}$$

and $\ell_{x,y}(i)$ is the length of a longest substring of y that matches a substring starting at position i of x. Note that d_ℓ is still not a metric, since it violates triangle inequality, however, it does constitute one more measure explicitly based on the substring compositions of x and y. Note also that the average length of maximal common substrings adopted here is a coarser measure than the individual substring frequencies used by d_{cos} and d_{js}.

Kolmogorov's approach to information [87] may also lead to measures of mutual compressibility. Recall that the Kolmogorov complexity $K(x)$ of a string x is the length of the shortest program that outputs x and halts on a fixed universal machine, while the conditional Kolmogorov complexity $K(x|y)$ is the length of the shortest

program that computes x if y is given in input. Li et al. [93] showed that the *normalized information distance* $d_{nid}(x,y) = (\max\{K(x|y), K(y|x)\})/(\max\{K(x), K(y)\})$ satisfies the properties of a metric up to an additive term, and is *universal* in that $d(x,y) \leq f(x,y)$ for any upper-semicomputable normalized distance f, up to an additive term. Since K is not computable [87], neither is $d_{nid}(x,y)$, which is approximated in practice by the *normalized compression distance*:

$$d_{ncd}(x,y) = \frac{\max\{C(x|y), C(y|x)\}}{\max\{C(x), C(y)\}} \approx \frac{\min\{C(x \cdot y), C(y \cdot x)\} - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}} \tag{1}$$

where $C(x)$ is the length of the output of a mundane compressor on input x, and symbol $\cdot$ denotes concatenation. Most applications of d_{ncd} to the classification of texts use Lempel-Ziv-like compressors [90, 185], thereby making d_{ncd} an implicit measure of the substrings shared by the LZ77 dictionaries of the two strings [153], albeit probably perturbed by the heuristics of the compressor.

2.1.2 Gapped patterns in phylogeny

Patterns with gaps are a successful formalism to represent structural and functional information in biological sequences: for example, most of the signatures in biologically significant databases like PROSITE [155] contain gaps with fixed lengths [74, 81], and algorithms for the automatic extraction of gapped motifs in many flavors have flourished (see, e.g., [81, 124] and references therein). Due to their ability to recapitulate all motifs that occur in a string, maximal motifs have attracted a fertile line of research [30, 67, 137]. Like patterns in manually curated databases, however, maximal motifs extracted by unsupervised algorithms have mostly been applied to build *signatures* of protein families, with a range of complexity that goes from a single motif to sets of motifs that occur with variable order, multiplicity and position [43, 102, 103, 160].

Recall that by *composition* we mean the set of all structures of a given type that

occur in a dataset. The transition from patterns seen as signatures to comprehensive compositional studies probably started with the unsupervised extraction of all maximal motifs with given density bounds and support from the GenPept database [138], with the aim of building a dictionary of all maximal motifs that occur in any known protein sequence. Correlating such building blocks to structure and function provides a way to understand protein organization, thereby enabling a pipeline for the unsupervised functional and structural annotation of proteomes [140]. Motifs in the dictionary have been shown to contain information at multiple levels of abstraction: some motifs are specific to a protein family, others are specific to a phylogenetic taxon, and yet others cross protein families and phylogenetic groups, suggesting themselves as universally reused gapped modules that resonate with solid ones identified earlier [71, 72]. The very idea of relating motif composition to phylogeny probably surfaces for the first time in this dictionary, albeit being still seen from a signature viewpoint: the authors of [138] ask for the set of motifs that characterizes a specific clade, that are shared among a given set of clades, and that occur in all known clades, and they provide examples of motifs that are archaea-specific, bacteria-specific, shared between archaea and bacteria and between archaea and eukaryotes. A systematic study on the classification power of gapped motifs is however deferred. The dictionary of motifs was subsequently recompiled using the proteomes of 4 archaea and 13 bacteria [139]: once again, motifs are used for functional annotation and as signatures of protein families, and the composition of motifs is not compared across the two clades.

The notion of composition vector based on normalized counts of occurrences of gapped patterns looms already in the few other large-scale compositional studies on gapped patterns. These studies systematically collected all occurrences of PROSITE's regular expressions in the translated intergenic regions of the fly, yeast and human genomes [184], and in a set of 42 proteomes [116], respectively, exploiting existing theoretical tools to compute expectation and variance (see e.g. [21, 117] and references

therein). Both studies revealed subsets of patterns to be overrepresented and other subsets to be underrepresented, showing functional preference in proteomes, and discovering, in intergenic regions, relics of ancient proteins that have been deactivated by accumulated mutations. Such compositional preferences, however, were not used to build phylogenies. Composition vectors based on various notions of gapped patterns have then been extensively built by the string kernel community as a prerequisite to classify biosequences using the SVM machinery. For a small sampler, we mention here vectors containing the raw frequency of motifs in the eBLOCKS database [22] – rigid gapped patterns with substitution groups extracted in an unsupervised way from the SwissProt database using the eMOTIF heuristic –, vectors containing the frequency of all maximal rigid gapped motifs with high density occurring in the dataset [48], vectors indexed by all possible strings in $(\Sigma \cup \{\bullet\})^*$ with k solid characters and at most m gaps [91], vectors indexed by k-mers, but containing the number of occurrences of each k-mer as a subsequence with prescribed number and length of gaps [91, 104, 148], and vectors indexed by all possible pairs of spaced k-mers [101]. These studies, however, applied composition vectors to the task of discriminating between the biological sequences belonging to a class (e.g. a node in the SCOP tree or a group of enzymes) and those not belonging to a class, rather than to reconstructing hierarchical clusters or entire phylogenies. Perhaps the efforts in this line of research that came closer to the reconstruction of phylogenies were the use of the normalized frequency of short, dense, gapped maximal motifs to detect horizontal gene transfer events [165] and to classify variable-length DNA fragments coming from several metagenomes [109]. In this latter work, a hierarchy of multiclass support vector machines was used to discriminate the members of a phylogenetic taxon from those not belonging to that taxon, at the domain, phylum, class, order and genus level.

To date, few phylogenies inferred from gapped motifs exist, but none of them compares the motif *composition* of biosequences explicitly. To compute the distance

between two sequences x and y, the authors of [77] concatenate the realizations of all rigid, gapped, maximal motifs that occur exactly once in both x and y, forming two new sequences x', y' of the same length. A conventional maximum-likelihood estimate based on a model of character evolution is then used to compute the distance between x' and y'. This methodology effectively uses rigid gapped motifs as anchors for local alignments, comparing the characters that fill corresponding gaps in two sequences rather than the repertoire of motifs in the sequences. Moreover, motifs with multiple occurrences in the same sequence are systematically discarded. Motifs with *flexible* gaps are used to classify mitochondrial genomes in [12], but inside the algorithmic information framework of the Normalized Compression Distance [37]: the distance between two strings depends here on their mutual compressibility with a greedy offline compressor that iteratively shrinks the pair using the motif that yields the best gain, possibly in lossy mode [16]. The motif composition of the two strings is thus compared only implicitly.

As mentioned, the present chapter investigates also how classification quality varies when moving from all motifs in a family to compact subsets capable of generating the whole family. In rigid gapped motifs, the notion of using a basis to characterize and compare strings without resorting to alignments originated with the very definition of such bases [127]. However, few alignment-free methods study similar issues of minimality. SVD is the typical dimension-reduction and denoising step after the construction of composition vectors [48, 163, 164], however the features of the resulting orthonormal basis have no clear interpretation as substrings or patterns. Elsewhere [40, 41, 42] it is conjectured that moving from distances based on *common rigid gapped motifs*, i.e. on rigid gapped motifs with at least one realization in two strings, to a non-redundant subset with no mutual dependency and capable of generating all common rigid gapped motifs, should improve the performance of kernel methods by removing redundancy. Competitive results are reported in the remote

homology detection of proteins, but the distortion on distance that this approach should be capable of avoiding is not quantified empirically nor formally.

Another objective of the present chapter is studying what happens to phylogenies when increasingly *sparser* motifs are used in composition vectors. Sparsity has been systematically penalized in string kernels, typically by weighting gaps with exponentially decreasing functions of their length or number [91, 104, 148]. On the other hand, applications of motif discovery have been extremely liberal with gaps (except rare exceptions, e.g. [81]), showing that sparse structures do carry biological information. Table 1 summarizes the densities used in a sampler of papers that extract maximal and elementary motifs (as defined in the following section) from biological sequences. Experiments with gapped string kernels have used comparable or even higher densities than those listed in the table (see e.g. [91]).

2.2 Phylogeny construction with rigid gapped motifs

Let Σ be a reference alphabet and let $\bullet \notin \Sigma$ be a don't care. Using standard notation, we define the following partial order among elements of $\Sigma \cup \{\bullet\}$: $a \preceq b$ iff either $a \in \Sigma$ and $b = \bullet$, or $a \in \Sigma$ and $b = a$. We also define the binary operator $\oplus$ on Σ as follows: $a \oplus b = a$ if $a = b$, and $a \oplus b = \bullet$ otherwise. In the present paper we will use the term *gapped pattern* (or just *pattern*) to denote any string in $\Sigma(\Sigma \cup \{\bullet\})^* \bullet (\Sigma \cup \{\bullet\})^*\Sigma$, i.e. any string in $\Sigma(\Sigma \cup \{\bullet\})^*\Sigma$ that contains at least one don't care. We will say that pattern v is a *subpattern* of pattern w if there is an index $i \in \{0, \ldots, |w| - |v|\}$ such that $w[i + j] = v[j]$ for all $j \in \{0, \ldots, |v| - 1\}$. We will use the term *gapped motif* (or just *motif*) to denote a gapped pattern that occurs at least two times in a string. With $\mathcal{L}_s(w)$ we will denote the set of occurrences of pattern w in string $s \in \Sigma^+$. Given an integer d, we will write $\mathcal{L}_s(w) + d$ to mean set $\{l + d \ : \ l \in \mathcal{L}_s(w)\}$.

The most natural way to introduce gaps in standard k-mers is probably the notion of *elementary motif*.

Table 1: Diachronic summary of papers that extract elementary and maximal gapped motifs from biological sequences. k solid characters can span a window of length at most h. n: minimum number of occurrences of a motif. o: homology allowed. Papers highlighted in gray detail the distribution of the number of motifs on density, length and support. Cursory hints at length and support can also be found in [103, 140, 166]. [51] considers only patterns with maximum length 10. [109] extracts patterns with length *exactly* h. [13] considers *flexible* motifs. [67], not included in the table, extracts maximal motifs with *global* density 0.65 and 0.8.

Ref.	Year	k	h	n	o	Dataset
[137]	1998	3 3	35 35	7 10		Core histone families H3, H4. Leghemoglobin family.
[138]	1999	6	15	2	•	All NCBI proteins.
[139]	1999	6	15	2	•	Translated ORFs from 13 Bacteria and 4 Archaea genomes.
[30]	2000	4 4	12 30	variable	•	Histone I protein family. GPCR protein superfamily.
[102]	2001	4 4 6 8	8 12 12 12	variable	•	GPCR protein superfamily.
[140]	2002	6	15	2	•	All SwissProt proteins.
[103]	2003	4	6	variable	•	Mammalian odor receptor proteins.
[43]	2005	4	8	2	•	Cupredoxin and multicopper oxidase protein families.
[48]	2006	3	6	10		SCOP families.
[77]	2006	4	16	2	•	Artificial polypeptides. Benchmark protein alignments.
[160]	2007	6	8	7		DNA upstream and downstream orthologous genes in Drosophila species.
[109]	2007	2 4 5	3 6 6	1		Metagenomic DNA: Sargasso sea, EBPR-sludge.
[51]	2007	2	3	variable	•	Enriched Eukaryotic Linear Motif datasets [65]
[13]	2010	2	3 4 6 7 12	variable	•	PROSITE families

Definition 10 (Elementary motif [137]). *A rigid gapped pattern w is a (k, h, n)-*elementary motif *of a string s if it has k solid characters, if it has length at most h, and if $|\mathcal{L}_s(w)| \geq n \geq 2$.*

Elementary motifs have strong ties to molecular biology: for example, self-contained "functional microdomains", believed to mediate 15-40% of all protein-protein interactions in intracellular signaling, are rigid gapped patterns with length at most 10 occurring in disordered regions on the surface of multidomain proteins [65]. The use of elementary motifs in phylogeny was probably hinted at for the first time in [77], and was then partially explored in [109].

Elementary motifs grow exponentially in k and h, thus limiting the values of these parameters that can be probed in practice. To handle longer and sparser structures, we turn to *maximal* motifs and their bases.

Definition 11 (Maximal motif [122]). *Let w be a pattern occurring at positions $\mathcal{L}_s(w) = \{i_0, i_1, \ldots, i_{n-1}\}$ in a string s, where $n \geq 2$. We say that w is* maximal in composition *if no other motif $v \neq w$ of s has $\mathcal{L}_s(v) = \mathcal{L}_s(w)$ and $v[i] \preceq w[i]$ for all $i \in \{0, \ldots, |w| - 1\}$. We say that w is* maximal in length *if no other motif $v \neq w$ of s is such that $|\mathcal{L}_s(v)| = |\mathcal{L}_s(w)|$ and w is a subpattern of v. We say that w is a* maximal motif *of s if it is both maximal in composition and maximal in length.*

Maximal motifs can grow exponentially in the length of the input string [122]. A first way to limit this explosion is imposing local density bounds.

Definition 12 (Dense maximal motif [137]). *A rigid gapped pattern w is a (k, h, n)-*maximal motif *of a string s if it is a maximal motif of s with $|\mathcal{L}_s(w)| \geq n$, and if every subpattern of w with exactly k solid characters has length at most $h > k$.*

As mentioned, a second way is to consider compact generators of the whole set of maximal motifs.

Definition 13 (Irredundant motif [122]). *A maximal motif w of a string s is* redundant *if there exist maximal motifs $w_0, w_1, \ldots, w_{n-1}$ of s such that $\mathcal{L}_s(w) = \bigcup_{i=0}^{n-1} \mathcal{L}_s(w_i)$. We call* irredundant *a maximal motif of s that is not redundant.*

Definition 14 (Tiling motif [126]). *A maximal motif w of a string s is* tiled *is there exist maximal motifs $w_0, w_1, \ldots, w_{n-1}$ of s ($w_i \neq w \; \forall \; i$) and integers $d_0, d_1, \ldots, d_{n-1}$ such that $\mathcal{L}_s(w) = \bigcup_{i=0}^{n-1} \mathcal{L}_s(w_i) + d_i$. We call* tiling *a maximal motif of s that is not tiled.*

The set of irredundant (respectively, tiling) motifs with at least n occurrences in a string s, together with their occurrence lists, contains sufficient information to generate any other maximal motif with at least n occurrences in s, together with its list, without knowing s itself [122, 125]. It is thus standard to call this set a *basis*. For $n = 2$, the size of the irredundant (respectively, tiling) basis is bounded by a quadratic (respectively, linear) function of the length of s [126]. The tiling basis is a subset of the irredundant basis, as well as of another distinguished set of maximal motifs that we include in our analysis.

Definition 15 (Autocorrelation [19]). *For strings x and y in Σ^+, let $w = x \oplus y$ be the string $w \in (\Sigma \cup \{\bullet\})^{\max\{|x|,|y|\}}$ such that $u[i] = x[i] \oplus y[i]$ for all $i \in \{0, n-1\}$ (we assume $x[i] = \bullet$ for $i < 0$ and $i \geq |x|$, and $y[i] = \bullet$ for $i < 0$ and $i \geq |y|$). Furthermore, given a string $w \in (\Sigma \cup \{\bullet\})^+$, we denote with $[w]$ the* pattern *obtained by removing all leading and trailing don't cares from w. A pattern w is an* autocorrelation *of a string s if $w = [s \oplus suf_i]$, where suf_i is the suffix of s starting at position $i \in \{1, |s| - 1\}$.*

To date, irredundant and tiling motifs have been used as guides for the alignment of multiple sequences [121], as codewords for lossy, as well as lossless, compression of texts [16] and images [3], and as features of string kernels for protein classification [41, 42].

As mentioned, we want to study how classification quality depends on the composition of autocorrelations and of elementary, maximal, irredundant and tiling motifs of a string. More specifically, unlike previous studies that assessed the performance of tree construction algorithms on few phylogenetic trees or tried to settle specific controversies in phylogeny, we want to produce results that are independent of the specific set of organisms used. However, we are not interested in artificial sequences generated by models of sequence evolution. We thus set the 2329 metazoan mitochondrial proteomes available from NCBI on June 2011 as our dataset $\mathcal{P}$, and we set the corresponding NCBI taxonomy $\mathcal{T}$ as our reference taxonomy. Mitochondria strike a good balance between phylogenetic significance and manageable string length: datasets containing few dozens mitochondria have been used repeatedly to assess the effectiveness of phylogeny reconstruction algorithms [12, 92, 164, 167].

Given a string $x \in \mathcal{P}$, we denote with $\mathbf{X}^{(e,k,h)}$ the corresponding composition vector indexed by all possible patterns with exactly k solid characters and length at most h. The component of $\mathbf{X}^{(e,k,h)}$ associated with pattern w contains the number of occurrences of w in x, normalized to the maximum possible number of occurrences of w in x, if w is a $(k, h, 2)$-elementary motif of x, and zero otherwise. For practical limits we set $h = 20$ and $k \in \{2, 8\}$, allowing a density than is approximately seven times smaller than the smallest density examined in previous studies on elementary motifs (Table 1). We will thus use the shorthand $\mathbf{X}^{(e,k)}$ for $\mathbf{X}^{(e,k,20)}$. Note that increasing k corresponds to the standard alignment-free methodology of increasing the length of substrings. Elementary motifs with the same k can however span different lengths, and thus have different densities. Similarly, we denote with $\mathbf{X}^{(m,k,h)}$ the composition vector indexed by all possible patterns w such that every substring of w that contains k solid characters spans at most h positions. The component of $\mathbf{X}^{(m,k,h)}$ associated with pattern w is zero if w is not a $(k, h, 2)$-maximal motif of x, and equals the normalized frequency of w in x otherwise. To render our experiments feasible, we are forced to

impose $k = 2$ and $h = 50$: this allows a local density that is approximately two times smaller than the smallest local density considered in previous applications of maximal motifs (Table 1), and captures all maximal motifs with 7 or more solid characters that occur in $\mathcal{P}$. This constraint is also permissive enough to match approximately 98.5% of all gaps contained in release 20.75 of PROSITE [30, 74]. We will thus use the shorthand $\mathbf{X}^m$ for $\mathbf{X}^{(m,2,50)}$. Finally, we set $\mathbf{X}^i$ (respectively, $\mathbf{X}^a$ and $\mathbf{X}^t$) to denote the composition vectors indexed by all possible patterns w, and containing at component w the normalized frequency of w in x if w is an irredundant motif (respectively, an autocorrelation or a tiling motif) of x, zero otherwise. Autocorrelations, irredundant and tiling motifs do not grow too fast in practice, thus we do not force any density constraint on them.

We are interested in studying how the quality of the reconstructed tree depends on the density of motifs (the ratio between the number of solid characters and length) and on their statistical significance. Given a composition vector $\mathbf{X}$, we denote with $[\mathbf{X}]_{d_0,d_1}$ the projection of $\mathbf{X}$ onto the subspace of patterns with density between d_0 and d_1 (inclusive). We measure the significance of seeing a pattern w occurring n times in a string x with the z-score of n, assuming that x has been generated by a Markov chain of order 1 whose transition probabilities match the empirical frequency of dimers in $\mathcal{P}$. We denote with $\langle \mathbf{X} \rangle_{z_0,z_1}$ the projection of $\mathbf{X}$ onto the subspace of patterns with z-score between z_0 and z_1 in x (inclusive). Given a set $P_i \subset \mathcal{P}$, we denote with $\mathcal{T}_i$ the corresponding subtree of the reference $\mathcal{T}$, and with $[T_i]^{e,k}_{d_0,d_1}$ (respectively, with $[T_i]^{m}_{d_0,d_1}$, $[T_i]^{i}_{d_0,d_1}$, $[T_i]^{a}_{d_0,d_1}$, $[T_i]^{t}_{d_0,d_1}$) the tree built from the strings in P_i as follows: first, we project each string $x \in P_i$ into the corresponding composition vector $[\mathbf{X}^{e,k}]_{d_0,d_1}$ (respectively, $[\mathbf{X}^{m}]_{d_0,d_1}$, $[\mathbf{X}^{i}]_{d_0,d_1}$, $[\mathbf{X}^{a}]_{d_0,d_1}$, $[\mathbf{X}^{t}]_{d_0,d_1}$); then, we build the matrix of pairwise Euclidean distances between each pair of such vectors; finally, we run neighbor joining on the resulting matrix. $\langle T_i \rangle^{e,k}_{z_0,z_1}$, $\langle T_i \rangle^{m}_{z_0,z_1}$, $\langle T_i \rangle^{i}_{z_0,z_1}$, $\langle T_i \rangle^{a}_{z_0,z_1}$,

$\langle T_i \rangle^t_{z_0,z_1}$ have similar definitions for z-scores. We are mainly interested in what happens at the two extremes of the density and z-score spectra. To study such extremes in elementary motifs, we take 100 random samples $P_0, P_1, \ldots, P_{99}$ with replacement from $\mathcal{P}$, such that each P_i contains the proteomes of 32 different organisms[1], and we plot the functions:

$$\begin{aligned}
\overrightarrow{d_{e,k}}(\delta) &= \frac{1}{100}\sum_{i=0}^{99} rf([T_i]^{e,k}_{0,\delta}, \mathcal{T}_i) \\
\overleftarrow{d_{e,k}}(\delta) &= \frac{1}{100}\sum_{i=0}^{99} rf([T_i]^{e,k}_{\delta,1}, \mathcal{T}_i) \\
\overrightarrow{z_{e,k}}(z) &= \frac{1}{100}\sum_{i=0}^{99} rf(\langle T_i \rangle^{e,k}_{-\infty,z}, \mathcal{T}_i) \\
\overleftarrow{z_{e,k}}(z) &= \frac{1}{100}\sum_{i=0}^{99} rf(\langle T_i \rangle^{e,k}_{z,+\infty}, \mathcal{T}_i)
\end{aligned}$$

where $rf(T_0, T_1)$ is the Robinson-Foulds distance [144] (abbreviated with RF in what follows) between trees T_0 and T_1, ranging in this case from 0 to 58. For studying autocorrelations, maximal, irredundant and tiling motifs, we similarly sample $\mathcal{P}$ and define the homologous functions $\overrightarrow{d_\alpha}$, $\overleftarrow{d_\alpha}$, $\overrightarrow{z_\alpha}$, $\overleftarrow{z_\alpha}$, $\alpha \in \{m, i, a, t\}$. In what follows, we will call $\overrightarrow{d_\alpha}$ and $\overleftarrow{d_\alpha}$ *left-to-right* analyses, and $\overrightarrow{z_\alpha}$ and $\overleftarrow{z_\alpha}$ *right-to-left* analyses. $\overleftarrow{d_\alpha}$ approximates the average behavior of classification quality in $\mathcal{P}$ as progressively sparser motifs are added to an initial core of extremely dense ones. We expect that motifs under a given density threshold cease to carry phylogenetic information and start to be dominated by noise. Similarly, $\overleftarrow{z_\alpha}$ approximates the average behavior of classification quality in $\mathcal{P}$ as progressively less statistically significant motifs are added to an initial core of highly significant ones. We expect the large mass of motifs with low z-score to be plesiomorphic features dominated by noise, and the few motifs with extremely high z-score to be peculiarities of each taxon that are difficult to find in other organisms. Apomorphic features should intuitively be found at "intermediate"

[1] 32 is a good balance between computation time and realistic input size.

z-scores, sufficiently high to distinguish them from random occurrences and yet low enough not to be idiosyncrasies of a given proteome.

The purpose of this section is not achieving the best classification in a specific dataset, but studying the shape of $\overrightarrow{d_\alpha}$, $\overleftarrow{d_\alpha}$, $\overrightarrow{z_\alpha}$ and $\overleftarrow{z_\alpha}$ on a large number of samples. This is why we have selected the components of our pipeline to maximize speed. For example, unlike state-of-the-art alignment-free algorithms, we do not store z-scores in composition vectors: this makes the computation of distance between two strings x and y extremely fast, because it allows to discard motifs that occur in neither x nor y, and to set to zero all components of $\mathbf{X}$ that correspond to motifs that do not occur in x: a crucial advantage when composition vectors are indexed by all possible rigid gapped patterns. Even *approximating* the z-score of seeing no occurrence in y of a motif that occurs in x makes our experiments unbearably slow. Removing z-scores from composition vectors has the additional advantage of avoiding the comparison among the z-scores of different motifs that would be implicit in the resulting distance: this comparison is unreliable in cases, like ours, in which the z-scores of motifs do not follow a normal distribution [158]. Finally, the size of the alphabet, the number of motifs, their length and their sparsity, make considering Markov chains of order greater than one impractical.

We use the publicly available version of TEIRESIAS [137] to extract dense elementary and maximal motifs, and we build fast implementations of the algorithms in [17, 126, 157] to extract irredundant and tiling motifs, and to compute the z-score of rigid gapped patterns. We feed our distance matrices to the PHYLIP package [54] for building neighbor-joining trees and for computing RF distances.

2.2.1 Experimental results

2.2.1.1 Classifying with elementary motifs

The set of elementary motifs with k solid characters and length 20 contains as much phylogenetic information as its supersets for any value of k: indeed, $\overrightarrow{d_{ek}}$ is approximately flat for any k (Figure 28a). As density increases, the number of motifs per density decreases like a polynomial of low order, thus denser motifs do encode phylogenetic information themselves. As in standard k-mers, the number of solid characters is the main force behind classification quality. In elementary motifs, however, the smallest RF distance is achieved by $\overrightarrow{d_{e3}}$ and $\overrightarrow{d_{e4}}$, while increasing k to 5 or larger *degrades* classification. Allowing elementary motifs with 3 solid characters to span up to 50 positions continues to show a flat $\overrightarrow{d_{e3}}$ curve (Figure 28a, insert), implying that elementary motifs with exactly 3 solid characters and length 50 are not dominated by noise, but rather encode as much phylogenetic information as denser ones.

The right-to-left analysis confirms that the sparsest motifs at all values of k carry phylogenetic signal: adding them to denser motifs makes tree topology converge, does not degrade classification quality at $k < 5$, and even *reduces* RF distance at $k \geq 5$ (Figure 28b). Dense motifs, on the other hand, belong to two different categories. Those with $k \geq 5$ contain little phylogenetic information: classification quality is poor (or even null for $k \geq 7$) when such very dense motifs are considered, and it gradually improves when progressively sparser motifs are added. Since the composition of standard *ungapped* k-mers with $k \geq 5$ yields good classifications on the same dataset (Figure 33a), these trends suggest that the performance of k-mer methods crucially depends on words that occur just once, or that do not occur, in mitochondrial proteomes. Elementary motifs with $k < 5$ and length $k + 1$, on the other hand, achieve the global minimum of the corresponding $\overleftarrow{d_{ek}}$, which remains constant when progressively sparser motifs are added. Once again, the smallest RF distance is achieved by $\overleftarrow{d_{e4}}$. Finally, we note that for $k > 2$ most motifs occur just two or three times in each

string, so our distance measure between the composition vectors of two strings becomes effectively the Jaccard distance between the corresponding sets of elementary motifs.

Elementary motifs preferentially amass at the low end of the z-score spectrum: for example, approximately 95% of all elementary motifs with $k = 4$ have z-score at most 6. Counterintuitively, motifs with low z-score carry a strong phylogenetic signal for each k: $\overleftarrow{d_{zk}}$ *decreases* or remains constant when such motifs are included, reaching its global minimum around 0; $\overrightarrow{d_{zk}}$ decreases or remains constant when motifs with progressively higher z-score are added, until it reaches a global minimum when the bulk of all elementary motifs with low z-score have been included. Figure 28c details $\overrightarrow{d_{z4}}$ and $\overleftarrow{d_{z4}}$: curves for different values of k follow similar trends.

2.2.1.2 Classifying with maximal motifs

The distribution of maximal motifs with respect to density is quantized (Figure 31a): in particular, the densities d for which at least one motif exists follow a $1/d$ trend. This is due to the fact that most maximal motifs have between 3 and 5 solid characters and variable lengths. Significantly, the distribution of the number of motifs on density is based on a single module that is iteratively repeated and scaled (Figure 31b). Preliminary experiments show that this regular shape persists when proteomes are reshuffled, implying that it is a property of the density of characters rather than a regularity in mitochondrial sequences. Previous works have studied such a distribution in different datasets [139, 138], but none has considered densities smaller than 0.4 and sufficiently small bins to detect a quantization.

Maximal motifs are inherently *infrequent and sparse*: approximately 80% of all maximal motifs occurs 3 or 4 times, and approximately 80% of all maximal motifs have density smaller than 0.1 (Figure 31). Our distance measure between the composition vectors of two strings thus becomes effectively the Jaccard distance between

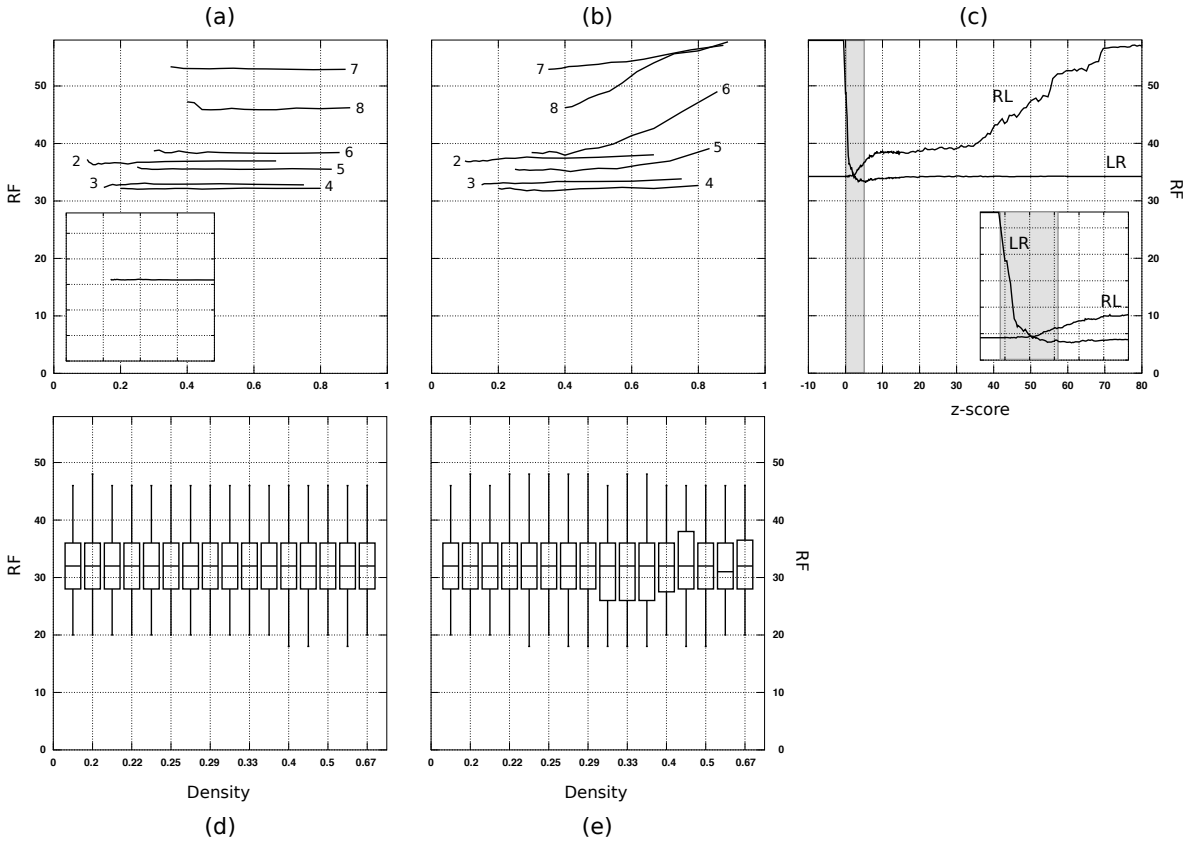

Figure 28: The classification quality of elementary motifs: $\overrightarrow{d_{ek}}$ (a) and $\overleftarrow{d_{ek}}$ (b) for $k \in \{2, \ldots, 8\}$. The insert in panel (a) shows $\overrightarrow{d_{e3}}$ at densities between 0 and 0.2 and at RF distances between 0 and 58. Panels (d) and (e) show the median, the 25th and 75th percentiles, the maximum and minimum of the RF distances of all samples for $k = 4$. To avoid clutter, the horizontal axis is distorted so that densities are equally spaced. (c) $\overrightarrow{z_{e4}}$ and $\overleftarrow{z_{e4}}$. Different values of k yield similar curves. The gray area indicates the approximate positions of the 5% and 95% values of the cumulative distribution of motifs, averaged over all strings in the dataset. The insert zooms the containing panel at z-scores between -2 and 10 and at distances 30 and larger.

the corresponding sets of motifs. At the high end of the density spectrum, maximal motifs cluster mainly around a small, discrete set of densities: 0.6, approximately 0.67, 0.75 and 0.8. $\overleftarrow{d_m}$ sharply decreases when motifs at these densities are progressively added, reaching a value that is just 2 units larger than the global minimum as soon as density 0.75 is reached (Figure 29b, insert). At this point, just 0.05% of all maximal motifs have been included. Using only motifs with density 0.8 is not sufficient to achieve the same RF distance, and using even denser motifs yields poor classifications. Counterintuitively, adding sparser motifs keeps $\overleftarrow{d_m}$ *constant or slowly decreasing*, and makes tree topology converge: the global minimum is reached at density approximately 0.135, when approximately 9% of all maximal motifs have been included (Figure 29b). Adding the remaining 91% motifs with even lower density causes only minor oscillations to $\overleftarrow{d_m}$, indicating that such motifs too are rich in phylogenetic signal, and that the taxonomy encoded by the composition of such sparse motifs agrees with the taxonomy encoded by the composition of denser ones. The high signal-to-noise ratio of extremely sparse motifs is confirmed by the left-to-right analysis: $\overrightarrow{d_m}$ uniformly *decreases* when progressively denser motifs are added, until it plateaus around density 0.15, when approximately 93% of all maximal motifs have been included (Figure 29a). The minima of $\overrightarrow{d_m}$ and $\overleftarrow{d_m}$ differ by just 1.3 units, suggesting that very sparse and very dense motifs tell similar phylogenetic stories, despite being such different structures: indeed, one set has average length 70 and average density 0.067, while the other has average length 5 and average density 0.8.

Motivated by the strong dependence between classification quality and number of solid characters k in elementary motifs, we perform the same analysis on maximal motifs. As above, let $\overrightarrow{d_{mk}}$ and $\overleftarrow{d_{mk}}$ be the curves of the left-to-right and of the right-to-left analysis of maximal motifs with exactly k solid characters. k is again a key factor in classification quality: quality improves going from $k = 2$ to $k = 3$ and 4, and degenerates for $k \geq 5$. $\overrightarrow{d_{m3}}$ and $\overrightarrow{d_{m4}}$ are approximately equal (Figure 29d), while $\overleftarrow{d_{m3}}$

is consistently the lowest in the right-to-left analysis (Figure 29e). Remarkably, the minima of $\overrightarrow{d_{m3}}$ and $\overleftarrow{d_{m3}}$ are approximately equal to those of $\overrightarrow{d_m}$ and $\overleftarrow{d_m}$, respectively, even though maximal motifs with $k = 3$ are just 34% of all maximal motifs. Of all maximal motifs with exactly 3 solid characters, the 79% with density at most 0.1 and the 50% with density at least 0.065 are sufficient to achieve the corresponding minima. In particular, using only motifs with $k = 3$ and density 0.6 or larger (approximately 4.5% of all maximal motifs with $k = 3$) leads to a classification quality that is just one unit larger than the global minimum of $\overleftarrow{d_{m3}}$ (Figure 29e, insert).

The distribution of maximal motifs on z-score is concentrated between scores approximately 0 and 25; unlike elementary motifs, it has a long decreasing tail at high z-score: approximately 10% of all maximal motifs in a proteome has z-score equal to 200 or larger. The right-to-left analysis shows that motifs with z-score equal to 100 or larger contain limited phylogenetic signal, as they need to be complemented by motifs with lower z-score to reach (at z-score one) the global minimum of $\overleftarrow{z_m}$, which is approximately 2.6 units larger than the global minimum of $\overleftarrow{d_m}$ (Figure 29c). In the left-to-right analysis, the bulk of motifs with z-score equal to 15 or lower contains sufficient phylogenetic signal to achieve the minimum of $\overrightarrow{z_m}$.

2.2.1.3 Classifying with motif bases

Consistent with previous studies [62][2], autocorrelations, tiling motifs and irredundant motifs are sparse, long and infrequent: 90% or more of these motifs have density 0.2 or smaller, and approximately 50% of all autocorrelations, 70% of all tiling motifs, and 40% of all irredundant motifs have length 100 or larger, compared to just 10% of all maximal motifs. Moreover, approximately 67% of all irredundant motifs, 90% of all autocorrelations and 99% of all tiling motifs occur 2 times, compared to just 4% of all maximal motifs (Figure 31). While the distribution of maximal motifs on length

[2]We thank Matthias Gallé for pointing out these distributional studies on tiling motifs, which inspired part of the present section.

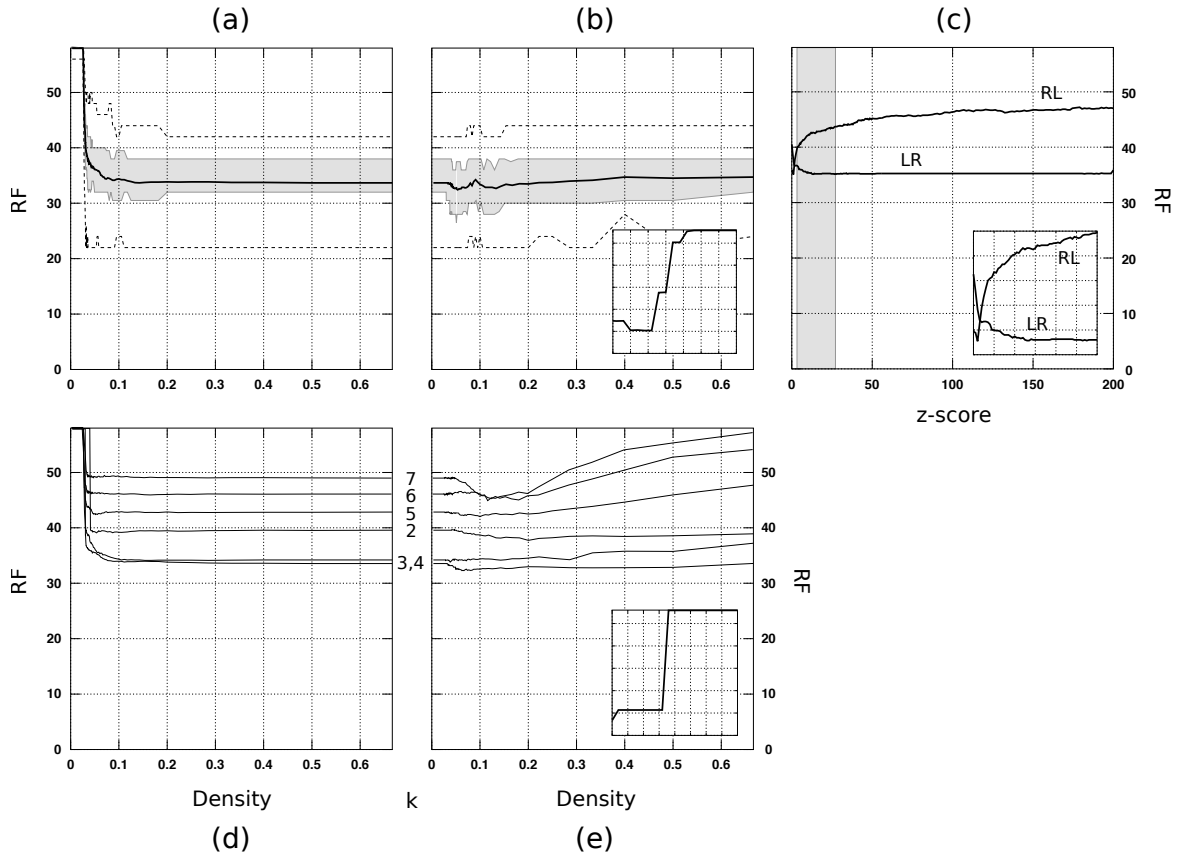

Figure 29: The classification quality of maximal motifs. $\overrightarrow{d_m}$ (a) and $\overleftarrow{d_m}$ (b) are represented with thick lines. The plots show also the first and third quartiles (gray areas), the minimum and maximum (dashed lines) of all samples taken. The insert in panel (b) shows $\overleftarrow{d_m}$ at densities 0.65 and larger and at RF distances 30 and larger. In the insert, horizontal grid lines occur every 5 units, and vertical grid lines occur every 0.05 units. Panels (d) and (e) show $\overrightarrow{d_{mk}}$ and $\overleftarrow{d_{mk}}$, respectively, for $k \in \{2, \ldots, 7\}$. The insert in panel (e) shows $\overleftarrow{d_{m3}}$ at densities 0.6 and larger and at RF distances 30 and larger. In the insert, horizontal grid lines occur every 5 units, and vertical grid lines occur every 0.05 units. Panel (c) shows $\overrightarrow{z_m}$ and $\overleftarrow{z_m}$. The gray area indicates the approximate position of the 25% and 75% values of the cumulative distribution of maximal motifs on z-score, averaged over all strings in the dataset. The insert zooms panel (c) at z-scores 30 and smaller and at RF distances between 34 and 44.

is unimodal, the distribution of autocorrelations and of irredundant and tiling motifs is *multimodal*, with peaks up to length 300 (Figure 31c): these shapes persist when proteomes are reshuffled, implying that they are not imputable to some regularity in the sequence. The distribution of irredundant motifs and autocorrelations on density is very similar to the distribution of maximal motifs: densities are again quantized, and the overall shape is based on a single module that is repeated and scaled.

As for maximal motifs, $\overrightarrow{d_a}$, $\overrightarrow{d_t}$ and $\overrightarrow{d_i}$ uniformly *decrease* when progressively denser motifs are added, and they finally plateau at a global minimum at density approximately 0.4 for irredundant and autocorrelations, and approximately 0.115 for tiling[3] (Figure 30a). This trend indicates that extremely sparse motifs do carry phylogenetic signal. Rather than remaining constant like $\overleftarrow{d_m}$, the curves of $\overleftarrow{d_a}$, $\overleftarrow{d_t}$ and $\overleftarrow{d_i}$ *decrease* when progressively sparser motifs are added, until they reach corresponding global minima at density approximately 0.115 for autocorrelations and irredundant, and approximately 0.15 for tiling[4] (Figure 30b). Significantly, such minima are smaller than the values of the functions at 0, indicating that *phylogenetic signal is differentially distributed along the density spectrum.* Indeed, the right-to-left analysis highlights a specific band of densities as more affected by noise: $\overleftarrow{d_a}$, $\overleftarrow{d_t}$ and $\overleftarrow{d_i}$ sharply increase when tiling motifs with density between 0.1 and 0.15 (approximately 28% of all tiling motifs), and autocorrelations and irredundant motifs with density between 0.085 and 0.115 (approximately 30% of the respective totals) are added (Figure 30b, insert). Including motifs with even lower density brings all curves back to values that are close to their global minima.

Despite having similar trends, the curves of autocorrelations, tiling and irredundant motifs differ considerably in absolute value. Tiling motifs display the worst performance: the distance computed using all tiling motifs is approximately 8.7 larger

[3]Including approximately 98% of all motifs in each basis.

[4]Including approximately 29% of all autocorrelations, 3% of all tiling motifs, and 36% of all irredundant motifs.

than both the distance computed using all autocorrelations, and the distance computed using all irredundant motifs, while the latter two differ by approximately 1.5 from each other. The curve of tiling motifs is consistently higher than the curve of autocorrelations at any density: $\overleftarrow{d_t}$ is at least 8 units larger than $\overleftarrow{d_a}$ in approximately 48% of the sampled densities, and $\overrightarrow{d_t}$ is at least 8 units larger than $\overrightarrow{d_a}$ in approximately 39% of the sampled densities. This indicates that the 90% *redundant* autocorrelations that are discarded during the construction of the tiling basis contain a strong phylogenetic signal. Another indication that redundancy is important in classification comes from irredundant motifs, a superset of the tiling basis that is approximately 15 times larger. Irredundant motifs display the best performance among the sets of motifs considered in this section: $\overleftarrow{d_i}$ is approximately 2 units smaller than $\overleftarrow{d_a}$ at its minimum, and the difference reaches peaks of 9 at higher densities.

Together with density and redundancy, the number of solid characters has again a strong influence on classification quality. As above, let $\overrightarrow{d_{ak}}$ and $\overleftarrow{d_{ak}}$ be the curves of the left-to-right and right-to-left analysis of autocorrelations with exactly k solid characters, and assume similar notation for tiling and irredundant motifs. In autocorrelations, distance decreases going from $k = 3$ to 4, then it monotonically increases for $k \geq 5$, both in the left-to-right and in the right-to-left analysis (Figure 30, c and d). A similar trend characterizes irredundant motifs, in which distance decreases going from $k = 2$ to 3, then monotonically increases for $k \geq 4$. Contrary to $\overleftarrow{d_a}$ and $\overleftarrow{d_i}$, no $\overleftarrow{d_{ak}}$ or $\overleftarrow{d_{ik}}$ increases when density decreases, thus the oscillations of $\overleftarrow{d_a}$ and $\overleftarrow{d_i}$ at low density are imputable to local changes in the abundance of motifs with different k.

Remarkably, using only irredundant motifs with exactly 3 solid characters (approximately 3% of the total) *improves* classification over using the whole set of irredundant motifs (Figure 30a): the minimum of $\overrightarrow{d_{i3}}$ is 5.4 units smaller than the minimum of $\overrightarrow{d_i}$, and it is reached using the approximately 80% sparsest fraction of all irredundant motifs with 3 solid characters; the minimum of $\overleftarrow{d_{i3}}$ is 2.2 units smaller than the

minimum of $\overleftarrow{d_i}$, and it is reached using the approximately 78% densest fraction of all irredundant motifs with 3 solid characters. $\overrightarrow{d_{i3}}$ improves by approximately 5 units over $\overrightarrow{d_i}$ at densities 0.4 and larger, and the difference reaches peaks of 12 at smaller densities. $\overleftarrow{d_{i3}}$ improves by approximately 4 units over $\overleftarrow{d_i}$ at densities 0.2 or smaller, with peaks of 10 around density 0.08. We stress that the minima of both $\overrightarrow{d_{i3}}$ and $\overleftarrow{d_{i3}}$ are achieved by long, sparse and infrequent motifs: the minimum of $\overrightarrow{d_{i3}}$ corresponds to a set of motifs with average density 0.09, average length 132 and average support 2.8; the minimum of $\overleftarrow{d_{i3}}$ is achieved by motifs with average density 0.14, average length 98, and average support 3.2. Such minima are equal, suggesting that motifs at the two ends of the density spectrum support similar phylogenies.

No value of k has a comparably distinguished role in autocorrelations. For example, using only autocorrelations with $k = 4$ (approximately 6% of the total) improves by just 1.7 over $\overrightarrow{d_a}$, and larger values of k degrade classification both in the left-to-right and in the right-to-left analysis.

2.2.1.4 Discussion

Rigid gapped motifs in polypeptides have traditionally been associated with *signatures* that group proteins into families with homologous function or structure. In this section we have shown that the *composition* of gapped motifs can be used to construct phylogenies from mitochondrial proteomes. Phylogenies with comparable distance to a reference taxonomy can be built using either extremely dense or *extremely sparse* motifs. For example, elementary motifs with exactly k solid characters and length 20 yield phylogenies of the same or better quality than those produced by elementary motifs with k solid characters and length $k+1$, and maximal motifs with density less than approximately 0.15 yield phylogenies of the same quality as those produced by maximal motifs with density 0.75 or larger. Using even lower densities degrades classification in maximal motifs and their bases, but surprisingly keeps groups of related

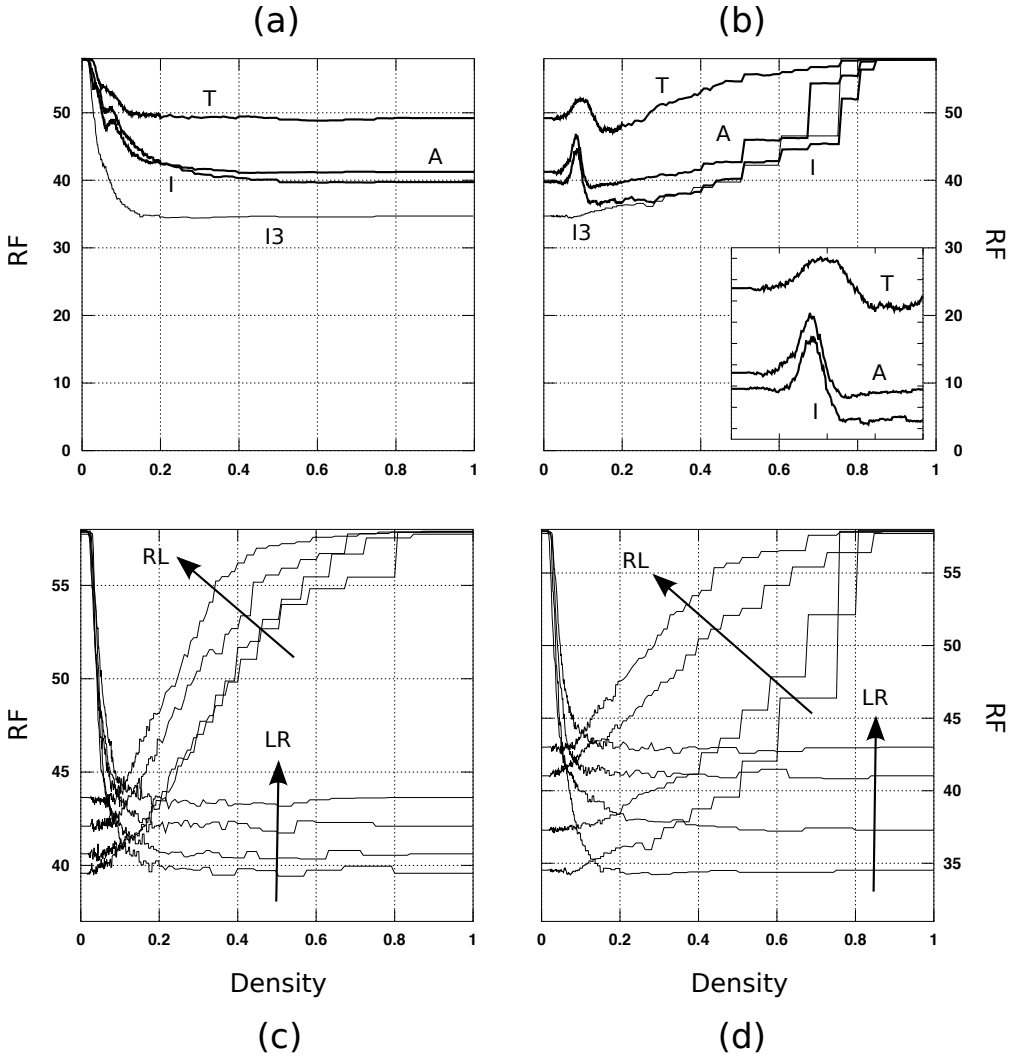

Figure 30: Classification quality of autocorrelations (A), tiling motifs (T), irredundant motifs (I), and irredundant motifs with exactly 3 solid characters (I3) as a function of density: left-to-right (a) and right-to-left (b) analysis. The insert in panel (b) zooms densities [0,0.2] and RF distances [35,53]. (c) $\overrightarrow{d_{ak}}$ and $\overleftarrow{d_{ak}}$ for $k \in \{4,5,6,7\}$. Arrows show the direction of increasing k. Curves for $k = 3$ are not shown because they are similar to the corresponding curves for $k = 5$. (d) $\overrightarrow{d_{ik}}$ and $\overleftarrow{d_{ik}}$ for $k \in \{3,4,5,6\}$. Curves for $k = 2$ are not shown because they are similar to the corresponding curves for $k = 4$.

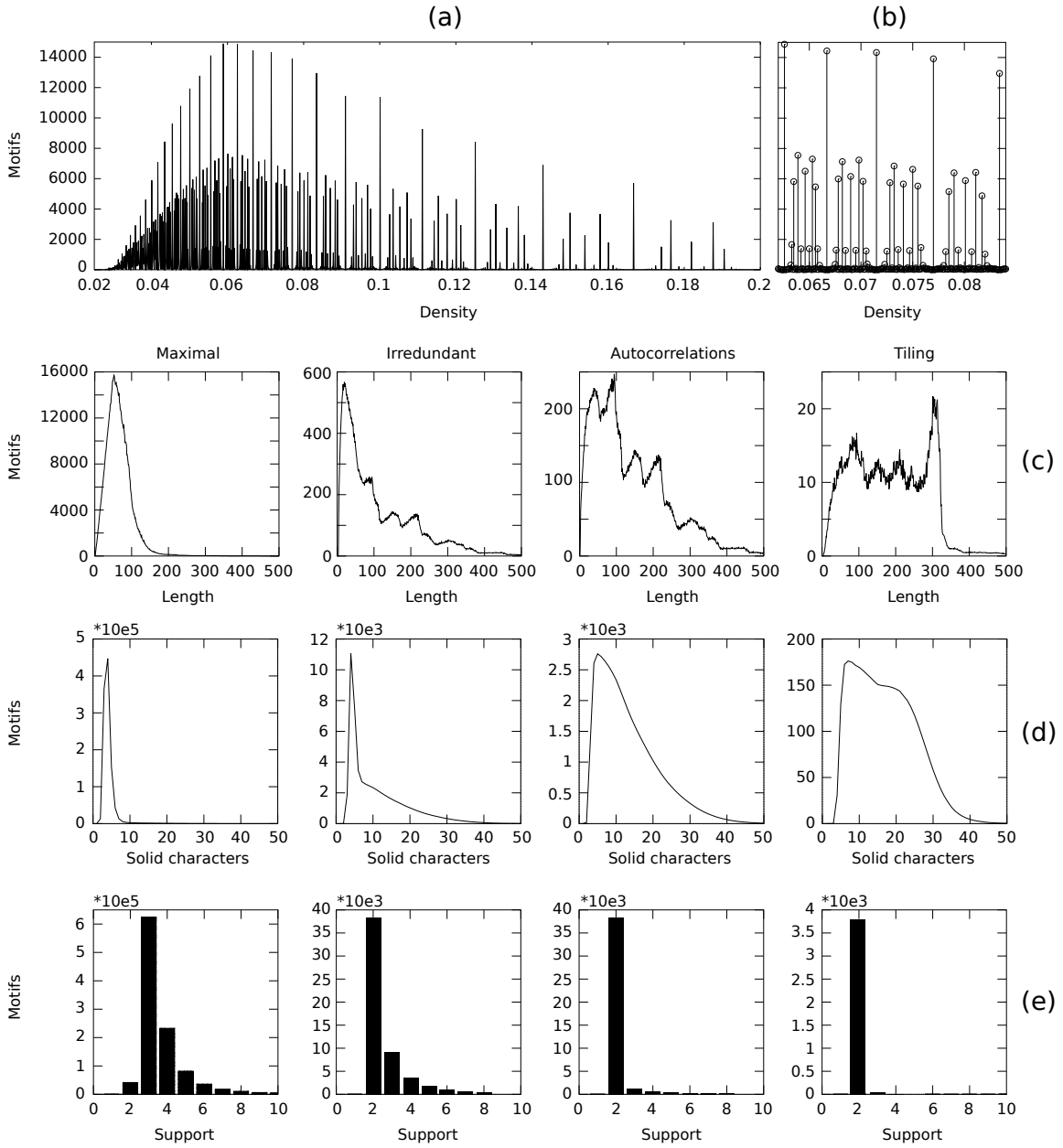

Figure 31: Density, length, number of solid characters and support in maximal motifs and their bases. (a) Number of maximal motifs with density d in the proteome of *Rattus norvegicus*. The same shape recurs in all strings of the dataset. (b) Detail of panel (a): the distribution of maximal motifs on density is based on a single unit that is repeated and scaled. (c,d,e) Average number of maximal, irredundant, tiling motifs and autocorrelations with length l (c), with k solid characters (d), and with support s (e) in the dataset.

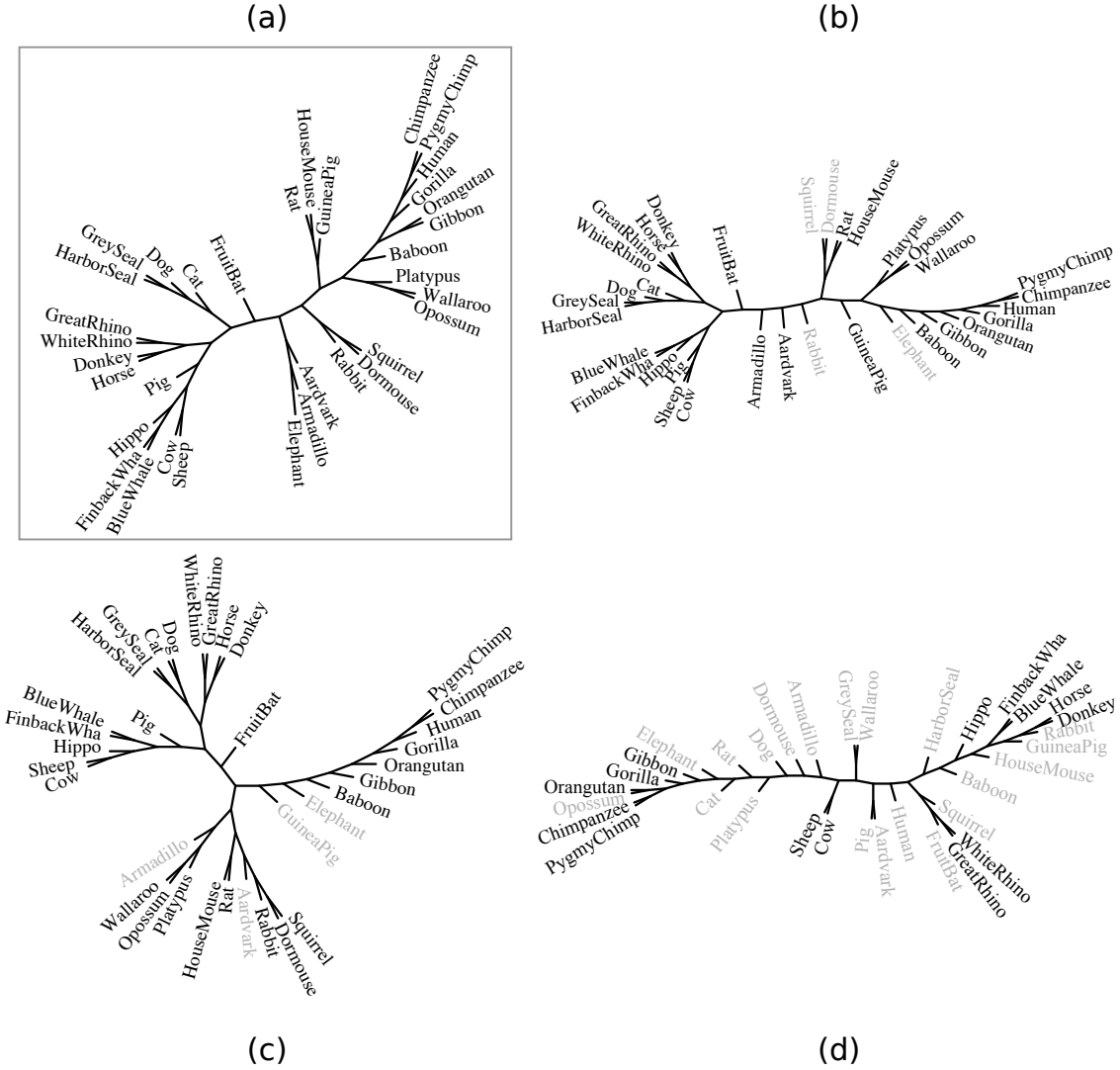

Figure 32: The composition of extremely sparse motifs carries a strong phylogenetic signal. (a) Reference tree from NCBI. See [167] and references therein for other algorithms applied to the reconstruction of this tree. (b) Tree built using *elementary* motifs with 3 solid characters and length 50. Average number of motifs per proteome: 30135. (c) Tree built using *maximal* motifs with 3 solid characters and density at most 0.0308. Average length of a motif: 98. Average number of motifs per proteome: 254. (d) Tree built using *irredundant* motifs with 3 solid characters and density at most 0.031. Average length of a motif: 118. Average number of motifs per proteome: 98.

organisms together (Figure 32). The length and sparsity of such motifs resonate in interesting ways with long-range correlations of various kinds that are known to have a key role in proteins [23, 173]: studying the structure of such sparse motifs and their occurrences in the protein space, as well as extending the alphabet of motifs to allow groups of homologous amino acids, would thus be natural extensions of this work.

In tiling motifs, irredundant motifs and autocorrelations, extremely dense motifs, as well as sparse motifs in a specific density range, contain comparatively little phylogenetic signal. Contrary to what has been observed in the remote homology detection of proteins [42], redundancy seems to be a key factor for the efficient reconstruction of phylogenies: classification quality *improves* when moving from the smallest tiling basis to its supersets, autocorrelations and irredundant motifs. Our analysis highlights also a third force behind classification quality: the number of solid characters. Contrary to the convergence seen when increasing the length of k-mers, classification with gapped motifs reaches its best at $k = 3$ or 4, and degenerates for larger k. In particular, considering only motifs with exactly 3 solid characters is sufficient – and sometimes even necessary – to achieve the best classification quality in elementary, maximal and irredundant motifs.

Another point in which our analyses differ from traditional k-mer approaches is the role of statistical correction. Downplaying k-mers with low statistical significance has been reported to be essential for achieving good classifications (see e.g. [36, 132]); our experiments, on the other hand, show that gapped motifs with z-score close to zero carry a strong phylogenetic signal, and classification quality *degrades* when such motifs are discarded.

Figures 33a and b summarize the sets of motifs that achieve the best average classification quality in our experiments. Such sets are extremely fast to compute in practice, and turn out to be largely disjoint from PROSITE. Remarkably, the average classification quality of such sets is comparable to state-of-the-art methods

based on substrings, even though all our motifs *repeat* at least two times in the input, and even though we use a simplistic setup based on Euclidean distance and raw frequencies (which in practice reduces to the Jaccard distance). This motivates further applications of gapped motifs to alignment-free sequence comparison, as well as a systematic search for the subsets of motifs that yield the best classification. The fact that substring-based and motif-based methods never push the average distance from the NCBI taxonomy below 30 suggests also the existence of a practical upper bound to the performance of alignment-free algorithms, that would be interesting to study more extensively.

Albeit being all at comparable distances from the NCBI taxonomy, the sets of gapped motifs shown in Figure 33a do not tell all the same phylogenetic story. As a first, qualitative glimpse into the problem of which motifs support which phylogeny, we computed the matrix of pairwise RF distances between trees produced by the best performing sets of gapped motifs and by a small sampler of substring-based alignment-free methods (CVTREE with lengths ranging from 3 to 7 [183], the Normalized Compression Distance using `gzip` [37], and the Average Common Substring[5] [167]), averaged over 100 random samples from our dataset (Figure 33c). The matrix shows at least two clusters: the first consisting of composition vectors with length greater than 3, NCD and ACS, the second consisting of elementary and maximal motifs with 3 solid characters. This suggests that phylogenies produced by elementary and maximal motifs are more similar to each other than to phylogenies built by substrings. Interestingly, composition vectors with length 3 tend to be more similar to the cluster of gapped motifs than to the cluster of substrings, while elementary motifs with 4 solid characters and length 20 tend to be more similar to the cluster of substrings than to the cluster of gapped motifs. Irredundant motifs seem to form a

[5]We thank David Burstein for providing an implementation of the Average Common Substring algorithm.

third cluster on their own, and ACS seems to be systematically different from all other substring-based methods. We leave to future research a more detailed study on this topic.

The fact that gapped motifs carry phylogenetic signal could be a peculiarity of proteomes (long regions without solid characters could represent loops where mutations are more likely, see e.g. [30] and [74]), or even just of *mitochondrial* proteomes. It is natural to envision experiments that apply gapped motifs to the reconstruction of phylogenies from the genic and intergenic DNA of longer genomes. Scaling to genomes would rule off the possibility of using the composition of *all* elementary and maximal motifs, and would move the focus on autocorrelations, tiling motifs, irredundant motifs, and on motifs with a controlled number of solid characters. Experiment with *flexible* gaps [13, 81] would also come natural: statistically significant maximal flexible motifs have already been shown to identify biologically significant patterns in PROSITE families [11].

2.3 Phylogeny construction with gapped LZW

The approach presented in this section explores the potential of the LZW compression algorithm – the variant of LZ78 proposed by Welch [175] – as well as of some of its recent lossy variants [6], in text classification and phylogeny reconstruction. Whereas LZW has a faster and simpler implementation than LZ77, the vocabulary underlying LZW is significantly smaller than that of LZ77. It seems thus natural, in presence of massive data, to inquire into the discriminating power of LZW. Specifically, we focus on recently introduced *gapped* variants of LZW that are equally straightforward to implement but allow for a controlled number of don't cares to be introduced in the substrings that compose the dictionary used during compression.

As is well known, the LZ77 encoding of a string s on alphabet Σ proceeds as follows: (1) initialize a dictionary to Σ; (2) assume to have encoded $s[0, i-1]$; let w

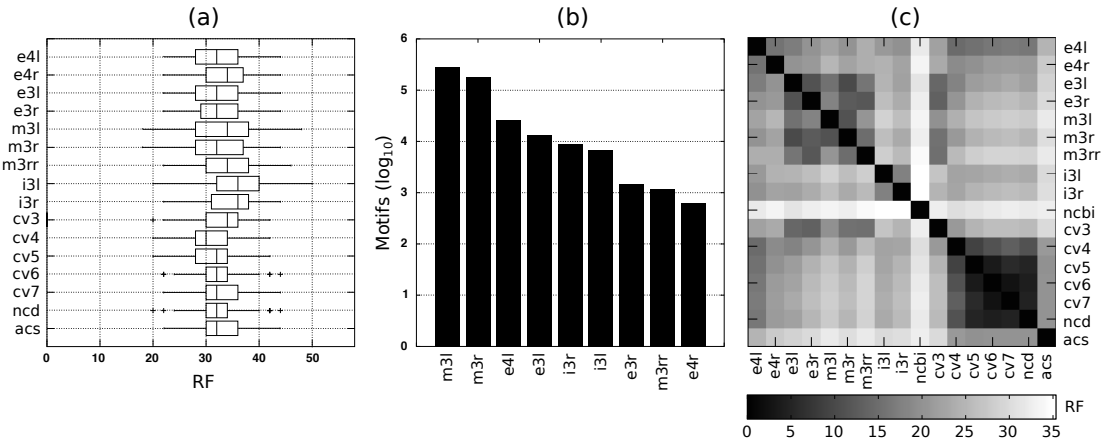

Figure 33: Average classification quality (a) and average size (b) of the sets of motifs that performed best in our experiments. Panel (a) shows median, 25th and 75th percentiles, minimum and maximum of RF distance over 100 samples of size 32 from $\mathcal{P}$. (c) Distance between the tree produced by a set of motifs and the tree produced by another set of motifs, averaged over 100 random samples from our dataset. `e4l`: elementary, $k = 4$, length 20. `e4r`: elementary, $k = 4$, length 5. `e3l`: elementary, $k = 3$, length 20. `e3r`: elementary, $k = 3$, length 4. `m3l`: maximal, $k = 3$, density ≤ 0.1. `m3r`: maximal, $k = 3$, density ≥ 0.065. `m3rr`: maximal, $k = 3$, density ≥ 0.6. `i3l`: irredundant, $k = 3$, density ≤ 0.15. `i3r`: irredundant, $k = 3$, density ≥ 0.075; `ncbi`: NCBI taxonomy. For reference, we include a small sampler of string-based alignment-free algorithms. `cvk`: composition vectors using k-mers. `ncd`: Normalized Compression Distance with `gzip -9`. `acs`: Average Common Substring.

be the longest prefix of $s[i, |s| - 1]$ that has an occurrence starting at some position $j < i$, and let $s[i + |w|] = a$; then append to the encoding the triplet $\langle j, |w|, a\rangle$, and repeat the process starting at $s[i+|w|+1]$. The direct encoding of the triplet $\langle j, |w|, a\rangle$ requires $O(\log(i) + \log(|s|) + \log(|\Sigma|))$ bits, which can beset in practice the benefit of compression. In addition, the implementation of this parse takes $O(|s|)$ steps, but it is not straightforward. Other variants, such as LZ78 [186] and the corresponding version by Welch [175], take trivially linear time to implement, and use much shorter encodings at the expense of a reduced vocabulary buildup. The vocabulary of LZ78 is limited to the phrases used in the parse, so that the decoder need only be supplied with the ordinal number of each phrase followed by the new one-character extension, and, thanks to an ingenious look-ahead, LZW does not even need to specify this character [175]. At the generic iteration of LZW, w is a prefix of the portion of the text waiting to be encoded. With a the symbol following this occurrence of w, LZW proceeds as follows: if wa is in the dictionary then the next symbol is read, and this is repeated with segment wa replacing w. If, on the other hand, wa is not in the dictionary, then the dictionary index of w is appended to the output file, and wa is added to the dictionary; following this, w is reset to a and processing resumes from the symbol following a. Once w is initialized to be the first symbol of the source text, "w belongs to the dictionary" is established as an invariant in the above loop. The resulting set of codewords obeys the *prefix closure* property, in the sense that if a codeword is in the set, then so is also every one of its prefixes.

We focus here on variants of LZW that are equally fast to implement and admit for gaps to be interspersed with cleartext. In the gapped adaptation LZWA described in [5, 6], we maintain a dictionary of *patterns* $D_p \subset (\Sigma \cup \{\bullet\})^+$, where $\bullet$ represents a wildcard or "don't care" symbol that may take up one of several specifications, and a dictionary of *resolvers* $D_r \subset \Sigma^*$. The pseudocode in Figure 34 details the algorithm. The first part of LZWA is identical to LZW, except that LZWA seeks now the longest

phrase that can be reconstructed by shuffling a string in D_p with a string in D_r. At each iteration, the algorithm maintains a pair of current strings $w \in D_p$ and $w' \in D_r$: if $(wa \in D_p)$ or $(w\bullet \in D_p) \wedge (w'a \in D_r)$, then w and w' are updated and the process is repeated; otherwise, the ordinal numbers of w and w' are appended to the output and D_p and D_r are suitably updated. In particular, $w\bullet$ is added to D_p only if the node corresponding to w in the trie of D_p already has α children labelled by symbols in Σ. Note that the information needed for the shuffle is provided *implicitly*, since the gaps that the decoder will find in w can be filled with the characters of w' in exact succession. Note also that this algorithm is greedy: if we assume $\Sigma \cup \{\bullet\}$ to be sorted with $\bullet$ its maximum element, then its seek phase finds the lexicographically least phrase in D_p occurring at the current position. Looking for a *best* phrase, e.g., the one minimizing mismatches, is feasible but time consuming. Clearly, with respect to standard LZW, the encoding of a single phrase doubles in format and probably in size, suggesting that this variant achieves better compression only if there is a sufficiently high reuse of patterns, resolvers, or both.

The insertion of gaps in D_p can be controlled in two, not mutually-exclusive ways. The first one consists in explicitly prohibiting patterns with an excessively high number (or density) of gaps. The effect of this is to limit the size of D_p and D_r, but also to shorten the length of phrases. The second way consists in decreasing the arity of a node of D_p, which can vary from $|\Sigma|$ (as in the original LZW) to 1. A smaller arity pushes LZWA towards building patterns with a larger number of gaps (Figure 35), thereby shifting from D_p to D_r the information about the source; this gives D_r an increasingly larger control over the length of a match with the current dictionaries and ultimately on the size of the compressed file. In what follows, we study in particular the interplay between arity and classification performance in LZWA.

```
D_p ← Σ, D_r ← Σ
w ← first input character   // current pattern
w' ← ε   // current resolver
output ← ⟨code(w), code(w')⟩
repeat until no more input characters
    a ← next input character
    if wa ∈ D_p then w ← wa    // extension continues
    else if (w• ∈ D_p) ∧ (w'a ∈ D_r) then
        w ← w•,  w' ← w'a   // extension continues
    else   // extension impossible
        output ← output · ⟨code(w), code(w')⟩
        n ← node associated with w in the trie of D_p
        c(n) ← number of children of n labelled by a symbol in Σ
        if c(n) = α then
            if w• ∉ D_p then D_p ← D_p ∪ {w•}
            if w'a ∉ D_r then D_r ← D_r ∪ {w'a}
        else D_p ← D_p ∪ {wa}
        w ← a, w' ← ε
    end if
end repeat
```

Figure 34: Pseudocode for LZWA. α is a user-specified upper bound on the arity of a trie; ϵ denotes the empty string; $\bullet$ is a don't care; $\cdot$ is the concatenation operator.

2.3.1 Experimental setup

Given a string $s \in \Sigma^+$, let $C(s)$ be the output of LZWA on input s, let $D_p(s)$, $D_r(s)$ be the corresponding dictionaries of patterns and resolvers, and let $D(s) \in \{D_p(s), D_r(s)\}$ be any of the two dictionaries associated with s. We use $D^*(s)$ to denote the set of *maximal strings* in $D(s)$, i.e. the set of strings in $D(s)$ that are each not a prefix of any other string in $D(s)$. In the prefix relation, $\bullet$ is considered as an additional symbol that differs from all other symbols in Σ. Given another string $t \in \Sigma^+$, we use $D^*(s,t)$ to denote the set of maximal strings in $D(s) \cap D(t)$. Given a string $w \in D(s)$, we denote with $f_s(w)$ the number of times w is referenced in $C(s)$. We wish to compare the performance in classification with varying arity, based on a handful of measures that relate to the composition of the dictionaries. Specifically, we use the Jaccard metric d_j between $D(s)$ and $D(t)$, and its variant d_{j^*}, respectively defined by:

$$d_j(s,t) = 1 - \frac{|D(s) \cap D(t)|}{|D(s) \cup D(t)|} \qquad d_{j^*}(s,t) = 1 - \frac{|D^*(s) \cap D^*(t)|}{|D^*(s) \cup D^*(t)|}$$

We also use the cosine metric d_c between the vectors the components of which are all the strings in $D(s) \cup D(t)$ and taking values $f_s(w)$ and $f_t(w)$, respectively. Finally, we include the following measure:

$$\begin{aligned} d_h(s,t) \;\; = \;\; & \frac{1}{2} \cdot \frac{|D^*(s,t)| \log |t| + |D^*(s,t)| \log |s|}{\sum_{w \in D^*(s,t)} |w|} + \\ & -\frac{1}{2} \cdot \frac{|D^*(s)| \log |s|}{\sum_{w \in D^*(s)} |w|} - \frac{1}{2} \cdot \frac{|D^*(t)| \log |t|}{\sum_{w \in D^*(t)} |w|} \end{aligned}$$

an adaptation of the average common substring distance d_ℓ that focuses on the average length of maximal strings in $D(s) \cap D(t)$ (like the original measure, it is symmetric and it is zero when $s = t$; see Section 2.1). We further experiment with an approximation $C(s|t)$ of Equation 1, which uses the length of the string resulting from the compression of s when the dictionaries of LZWA are initialized to those of t. In the lossless compression of a string s, $C(s)$ takes into account the size of both the pattern

and the resolver parts of the compressed file; in lossy compression, it only measures the size of the pattern part.

We adopt the neighbor-joining implementation provided by the PHYLIP package [54] to build the trees resulting from our pairwise dissimilarity measures, and we measure the quality of classification in terms of the Robinson-Foulds distance between the trees resulting from our computations and a reference tree [144]. To give our measures some generality, we take a reference tree with few branches, each representing a macroscopic difference in a given domain. Specifically, we consider the following trees: $\mathcal{T}_1$ =(((Rodentia, Serpentes), (Araneae, Hemiptera)), (Ascomycetes, Basidiomycetes), (Magnoliophyta, Chlorophyta)), a subtree of the NCBI eukaryotic mitochondrial tree, reflects macroscopic evolutionary differences among eukaryota; $\mathcal{T}_2$ =((Linux, Apache), (Emacs Lisp, Scheme), (BLAS, EISPACK)), reflects macroscopic differences among programming languages, their dialects, and corresponding software projects; $\mathcal{T}_3$ =((Dante, Cavalcanti), (Ariosto, Machiavelli), (Capuana, Verga)), consists of writings of notable Italian authors separated by approximately 300 years from each other. For each leaf of a tree $\mathcal{T}_i$, we arbitrarily pick k strings belonging to the corresponding group (e.g. five mitochondrial proteomes from NCBI, the four longest source files in each software project, four writings from each author), and compile a dataset $\mathcal{S}_i$. Next, to estimate the classification performance of LZWA on $\mathcal{S}_i$, we build, for each arity α, approximately 50 random samples of $\mathcal{S}_i$ such that each sample contains two strings from each leaf of $\mathcal{T}_i$. We then summarize the RF distance between $\mathcal{T}_i$ and the tree reconstructed from each sample in a box plot[6]. We discuss $\mathcal{S}_1$ in some detail, and only mention the results produced with the other datasets.

Since we are interested here in classification more than in compression at all costs,

[6]Central marks represent medians and boxes mark the 25th and 75th percentiles. Whiskers extend to the most extreme data points not considered outliers, and outliers are plotted as individual circles. An "outlier" is a value larger than $p_{75} + 1.5(p_{75} - p_{25})$ or smaller than $p_{25} - 1.5(p_{75} - p_{25})$, where p_{25} and p_{75} are the 25th and 75th percentiles, respectively.

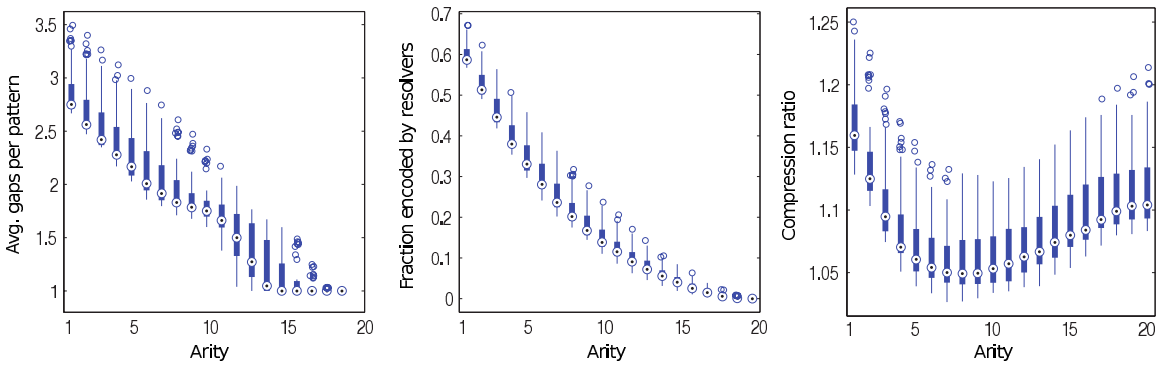

Figure 35: Dataset $\mathcal{S}_1$. (From left to right) Number of gaps per pattern, fraction of the original string encoded by resolvers, and compression ratio (size of the uncompressed file divided by the size of the compressed file) versus arity.

we build a clean implementation of LZWA with minimal space-saving heuristics. In particular, rather than using a sliding window, we keep the same pair of dictionaries throughout the entire compression process, and adjust the length of the codewords assigned to strings in a dictionary D to the corresponding frequency in the compressed file by using a suboptimal but fast $O(\log(|D|))$ heap.

2.3.2 Dynamics of patterns and resolvers

The gapped extensions of LZW liberally use the available resolvers irrespective of the strings they originated from. This tends to improve over the compression rate of standard LZW, but is antagonized by the need of two pointers per phrase. Figure 35 (right panel) illustrates this tradeoff with dataset $\mathcal{S}_1$: decreasing arity down to 8 or 9 increases the size of the compressed file, but from that point on the compressed file is smaller and smaller, until the compression of standard LZW is surpassed at arity 1 and 2. Datasets $\mathcal{S}_2$ and $\mathcal{S}_3$ exhibit a similarly concave plot, but with them the compression ratio of LZW is never surpassed by a gapped variant.

One may expect the smaller alphabet of LZW to degrade classification quality with respect to LZ77, and the reuse of resolvers in LZWA to blur our dissimilarity measures with respect to LZW, putting together sequences that are in fact distant. Our plots for dataset $\mathcal{S}_1$ confirm the first conjecture but show a different picture for

the second (Figure 36). The case in which $D_p(s) \approx D_p(t)$ and $D_r(s) \approx D_r(t)$ at the same time occurs only for few values of α. On the other hand, for every measure $d \in \{d_j, d_{j^*}, d_c, d_h\}$, there is an arity $\bar{\alpha}_d < |\Sigma|$ such that at all arities $\alpha \geq \bar{\alpha}_d$, the classification performance of d as applied to $D_p(s)$ and $D_p(t)$ is comparable to that of LZW. In $\mathcal{S}_1$, $\bar{\alpha} \approx 7$ or 8 for all dissimilarity measures on the dictionaries, *meaning that we can disregard approximately 20% of the original string (stored in D_r) and still get a classification comparable to the one based on* LZW. Performance degenerates on a roughly monotone slope when $\alpha < \bar{\alpha}$. Resolvers follow a complementary curve: classification performance is directly proportional to arity, it reaches a plateau of minimum efficiency when $\alpha \approx 12$ or 13, and at small arity it is comparable to standard LZW. This suggests that, when arity is high, information about the strings is mainly encoded by patterns, while resolvers are approximately random; when arity is low, information moves from patterns to resolvers. We remark that resolvers are strings that occur as *flexible* subsequences, whose flexibility and possible contexts are implicitly specified by patterns: these plots suggest that such flexible subsequences grasp information about the strings. We note that d_c and d_h do not perform significantly better than d_j, even though they incorporate the additional information on length and frequency, respectively. Combined with the fact that the performances of d_{j^*} and d_j are comparable, this suggests that *the set of maximal words in the dictionaries holds the key to classification* – an observation that resonates in interesting ways with recent methods that compare the maximal substrings of the input strings [7].

The performance of measures on D_p and D_r in classification does not allow us to predict the behavior of the normalized compression distance d_{ncd} (Equation 1). For example, assume that we compress string t using both $D_p(s)$ and $D_r(s)$ as starting points, and assume that $D_p(s) \approx D_p(t)$ and $D_r(s) \approx D_r(t)$. At high arity, the patterns in $D_p(s)$ are dense and very similar to those in $D_p(t)$, even though $D_r(s)$ and $D_r(t)$ might be uncorrelated. On one hand, this implies that the average length

of a string matching the dictionaries is large, so that the phrases in the compressed file tend to span long substrings and to induce a good conditional compression. On the other hand, the resolvers in $D_r(t) \setminus D_r(s)$ tend to have longer codes because they are seen after those of $D_r(t)$, and the few resolvers in $D_r(t) \cap D_r(s)$ have a frequency that does not reflect the one in $D_r(t)$: this injects noise in the resolvers part of the output. At low arity, the portion of the output reflecting D_p is affected by noise. Idiosyncrasies inherent to the specific implementation of the compressor affect d_{ncd} as well: for example, the length of the codewords in our compressed file only loosely reflects their frequency in the file itself. This introduces an additional component of noise that could worsen the quality of the classification. The plots of d_{ncd} show instead that *this measure is robust to both such sources of noise* (Figure 37): when only D_p or D_r is provided, the classification performance has a trend similar to measures on the corresponding dictionary; when both D_p and D_r are provided, this performance tends to improve at $\alpha < \bar{\alpha}$ and stays unchanged at $\alpha \geq \bar{\alpha}$.

We experiment also with two lossy variants of d_{ncd}; in both cases, we compute $C(s|t)$ by initializing $D_p(s)$ to $D_p(t)$, and take only the pattern part of the compressed file. In the first variant, $D_p(t)$ and $D_p(s)$ are built by maintaining also the corresponding $D_r(t)$ and $D_r(s)$; in the second, $D_r(t)$ and $D_r(s)$ are not maintained. The size of the resolver portion of the compressed file reflects the frequency of usage of the words of D_r during compression: by taking this size into account, we are implicitly measuring the similarity of the two frequency distributions. Therefore, we expect its removal not to be important at high arity, but to forfeit some information at low arity. The second variant works in a very similar way to the first: the only exception is that $D_r(t)$ and $D_r(s)$ are initialized to Σ^+ rather than to Σ: the two alternatives should therefore behave very similarly. Figure 38 confirms these conjectures: disregarding the resolvers portion of the compressed file induces the classification performance to degenerate earlier ($\alpha \approx 12$ versus $\alpha \approx 8$). The extent of this shift suggests that

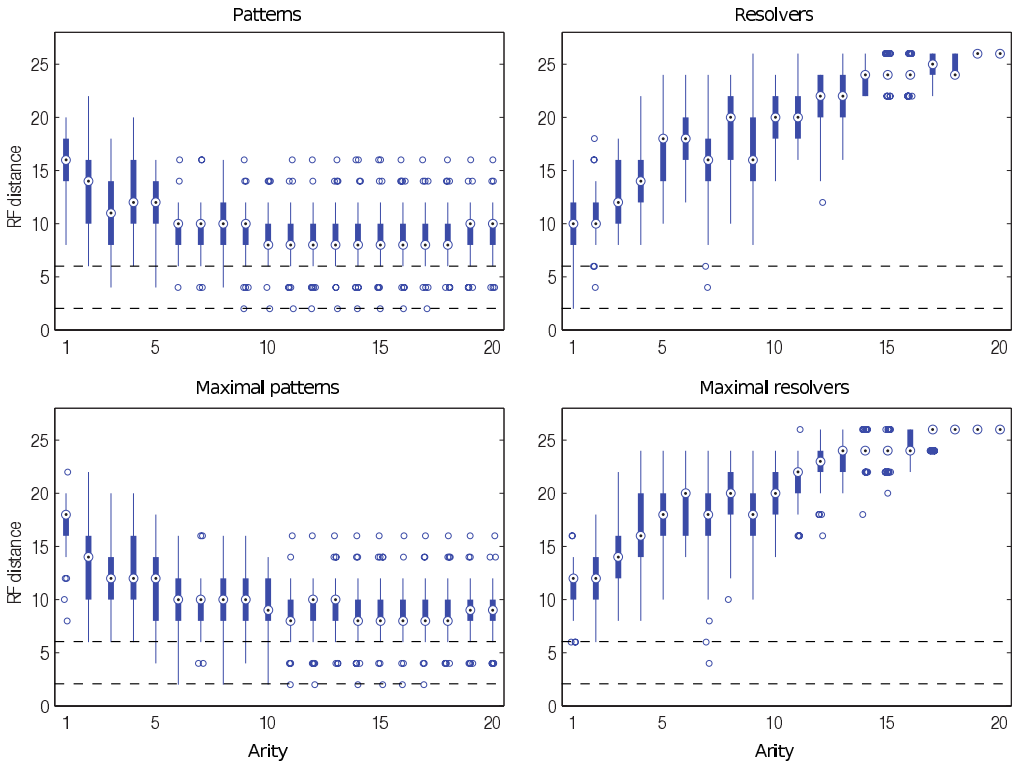

Figure 36: Classification performance of d_j (top) and d_{j^*} (bottom) on D_p (left) and D_r (right). Plots summarize 50 samples. Similar trends appear when plotting d_c and d_h (data not shown). In this and all the following plots, dotted lines indicate the 25th and 75th percentiles of the classification performance achieved by d_{ncd} when `gzip -9` is used as compressor.

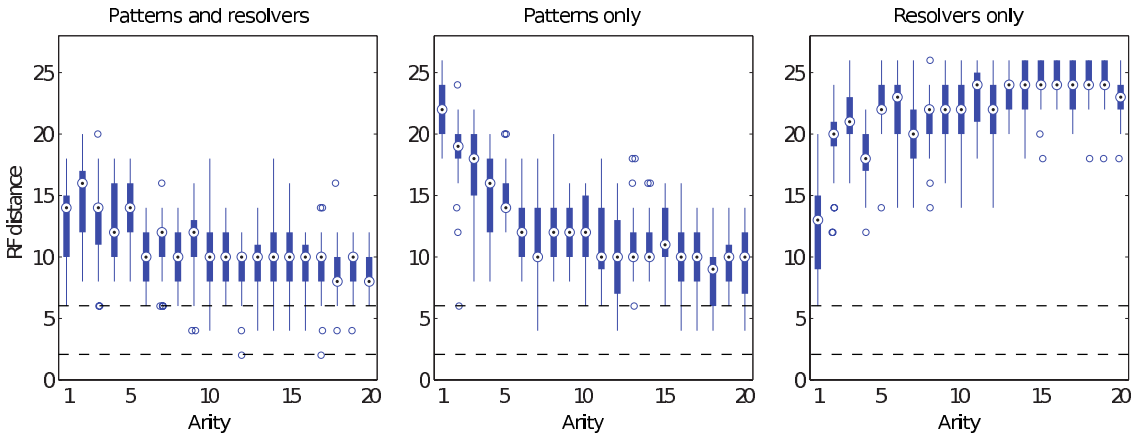

Figure 37: Classification performance of d_{ncd} when $D_p(t)$ is initialized to $D_p(s)$ and $D_r(t)$ is initialized to $D_r(s)$ (left); when just $D_p(t)$ is initialized to $D_p(s)$ (center); when just $D_r(t)$ is initialized to $D_r(s)$ (right). Plots summarize 36, 28 and 28 samples, respectively.

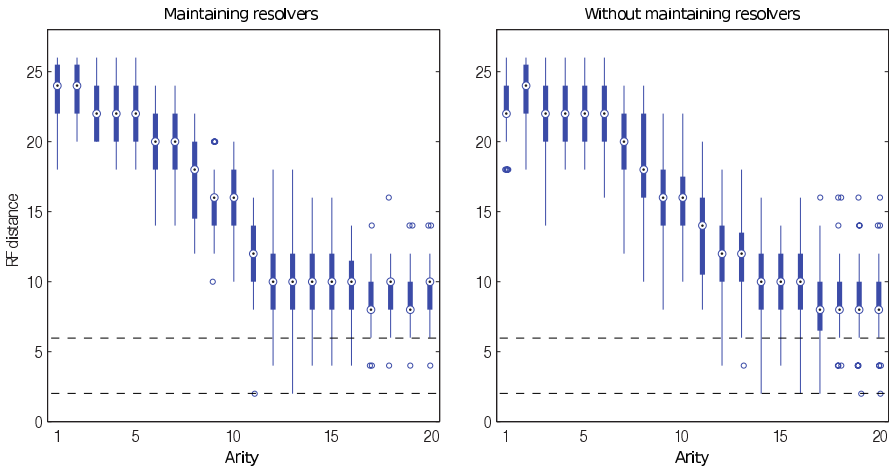

Figure 38: Classification performance of d_{ncd} with lossy compression. Plots summarize 32 and 39 samples, respectively.

this information is critical in d_{ncd}. By contrast, dictionary-based measures seem immune from this phenomenon. Significantly, even without maintaining D_r, these lossy measures can still classify with performances comparable to LZW at $\alpha \approx 15$.

The other two datasets $\mathcal{S}_2$ and $\mathcal{S}_3$ confirm the overall trends of patterns and resolvers, albeit exhibiting opposite performance behaviors. The dataset $\mathcal{S}_3$ of Italian literature is consistently difficult to classify (RF distance ≈ 12, maximum $= 16$) at any value of α. On the easier dataset $\mathcal{S}_2$ of programming languages, an almost perfect classification is achieved by all measures up to $\alpha \approx 10$, i.e. when up to approximately 10% of the original strings is discarded.

Finally, in order to assess the classification quality of measures on LZWA dictionaries on instances of practical significance, we abandon the toy datasets described above and experiment with $\mathcal{S}_4$, the set of 34 mammalian mitochondrial proteomes used by [167] and [93] to demonstrate d_ℓ and d_{ncd}, respectively. Specifically, we compare the trees produced by our measures on $\mathcal{S}_4$ with each one of the 13 maximum-likelihood trees produced by single common genes, as described in [167]. We find that d_h and d_j produce results that are largely comparable to, and in a number of cases better than, those of d_ℓ and d_{ncd} (Table 2 and Figure 39). Significantly, good results occur

Table 2: Distance between the trees constructed using d_j for varying arities (columns) and the maximum-likelihood trees produced by 13 proteins (rows). *Con*: majority consensus tree of the 13 single-protein maximum-likelihood trees. d_ℓ: see Section 2.1 and [167]. d_{ncd}: Equation 1 with a tuned DNA compressor [93], as reported in [167]. Light gray: $\min\{d_\ell, d_{ncd}\} \leq d_j \leq \max\{d_\ell, d_{ncd}\}$. Dark gray: $d_j < \min\{d_\ell, d_{ncd}\}$.

	10	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**Avg**	d_ℓ	d_{ncd}
A6	38	38	38	36	38	40	40	40	40	40	40	38.91	40	40
A8	40	38	42	44	42	44	44	44	44	44	44	42.73	42	42
C1	34	28	32	26	22	26	28	28	28	28	28	28.00	28	26
C2	42	42	38	40	38	40	38	38	38	38	38	39.09	42	40
C3	42	42	40	40	40	40	42	42	42	42	42	41.27	42	42
CB	40	34	32	30	28	28	28	30	30	30	30	30.91	30	24
N1	32	30	26	30	26	30	28	28	28	28	28	28.55	30	30
N2	26	20	30	26	18	22	18	22	22	22	22	22.55	24	24
N3	40	36	36	36	30	32	32	32	32	32	32	33.64	28	30
N4	34	30	30	32	30	30	30	30	30	30	30	30.55	28	24
NL	42	42	40	42	42	40	40	42	42	42	42	41.45	36	40
N5	30	26	26	22	14	14	8	14	14	12	12	17.45	18	18
N6	40	36	32	34	34	32	34	34	34	34	34	34.36	28	32
Con	30	28	26	26	14	22	18	22	22	22	22	22.91	16	18
Avg	36.43	33.57	33.43	33.14	29.71	31.43	30.57	31.86	31.86	31.71	31.71		30.86	30.71

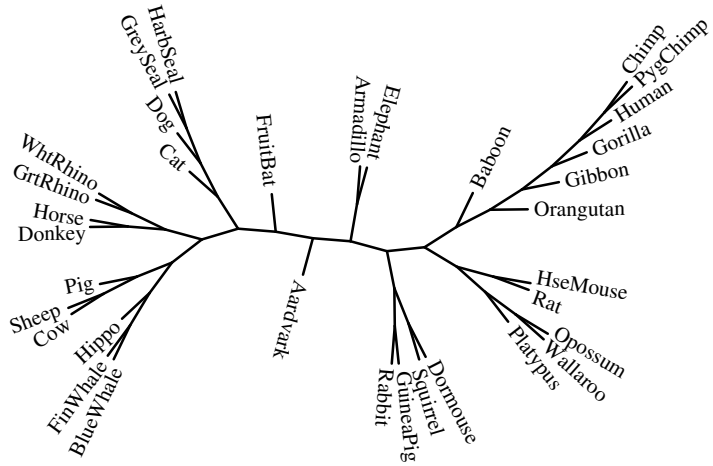

Figure 39: Dataset S_4, arity = 14. Neighbor-joining tree from measure d_j on the dictionaries of patterns.

for α ranging from 20 to at least 14, reinforcing the observation that D_p, even when impoverished by decreasing arity, still contains sufficient information to classify.

CHAPTER III

FASTER VARIANCE COMPUTATION FOR PATTERNS WITH GAPS

Determining whether a pattern is statistically overrepresented or underrepresented in a string is a fundamental primitive in computational biology and in large-scale text mining. In this chapter, we study ways to speed up the computation of the expectation and variance of the number of occurrences of a pattern with rigid gaps in a random string. Our contributions are twofold: first, we focus on patterns in which groups of characters from an alphabet Σ can occur at each position. We describe a way to compute the exact expectation and variance of the number of occurrences of a pattern w in a random string generated by a Markov chain in $O(|w|^2)$ time, improving a previous result that required $O(2^{|w|})$ time. We then consider the problem of computing the expectation and variance of the *motifs* of a string s in an IID text. Motifs are rigid gapped patterns that occur at least twice in s, and in which at most one character from Σ occurs at each position. We study the case in which s is given offline, and an arbitrary motif w of s is queried online. We relate computational complexity to the structure of w and s, identifying sets of motifs that are amenable to $o(|w| \log |w|)$ time online computation after $O(|s|^3)$ preprocessing of s. Our algorithms lend themselves to efficient implementations.

3.1 Introduction and state of the art

Given a string $w \in \Sigma^+$ and a random text $Z \in \Sigma^+$, the statistical properties of the occurences of w as a *substring* of Z have been extensively studied and repeatedly

applied to biological sequences (see e.g. [83, 105, 135] and references therein). Quantities of interest are typically the number of occurrences, the waiting time before the first occurrence, *r-scans* (the distance between an occurrence and the r-th next one), and corresponding quantities applied to higher-order structures, like *renewals* and *clumps* (maximal sets of overlapping occurrences). Traditionally, the focus has been on producing exact closed forms of the distribution and moments of such quantities or of corresponding asymptotic approximations, and bounds on approximation error.

Comparatively little is known about the *algorithmic* aspects of computing such quantities. For a string Z generated by an IID source, the expected value and variance of the number of occurrences of all prefixes of w in Z can be computed in overall $O(|w|)$ time, by embedding the computation in a landmark string searching algorithm for constructing the longest border of all prefixes of w [8]. This technique, combined with the linear-time construction of the suffix tree of a string s, allows to score and discover all significantly overrepresented and underrepresented substrings of s in overall $O(|s|)$ time if the measure of statistical significance f satisfies $w \equiv_R wx \Rightarrow f(w) \leq f(wx)$ for any $x \in \Sigma^+$, where $\equiv_R$ means right-equivalence in s [9, 10]. Similar dynamic programming schemes apply to strings with *mismatches*. The expected number of occurrences of a string w with up to k mismatches in an IID text Z can be computed in $O(k^2)$ time after a $O(k|w|)$ preprocessing of w [18]. A related algorithm allows to compute the expectation of all substrings of w with prescribed length in $O(k|w|)$ time, both for IID and for Markov sources [18, 128].

Measuring the statistical significance of strings with *gaps* or *don't cares* is a core primitive in computational biology [56]. A natural way to model gaps is requiring a pattern to occur as a *constrained subsequence* of a given text. Assume indeed that pattern w is a pair (w, D) in which $w = w_0 w_1 \dots w_{m-1} \in \Sigma^m$ is a string and $D = d_0 d_1 \dots d_{m-2} \in (\mathbb{N}^+ \cup \{\infty\})^{m-1}$ is a sequence of upper bounds on the distance between adjacent symbols in w. Fast approximations of the expectation and variance of the

number of occurrences of (w, D) in an IID text Z have been derived using standard properties of generating functions and the Chomsky-Schützenberger algorithm [58]. A related problem consists in computing the expectation and variance of the number of (possibly overlapping) *windows* of Z of size σ that contain at least one occurrence of string w as a subsequence. Setting $p(w, \sigma)$ to the probability that w occurs in a string of length σ, it is easy to see that $p(w, \sigma) = (1 - \mathbb{P}(w[|w| - 1])) \cdot p(w, \sigma - 1) + \mathbb{P}(w[|w| - 1]) \cdot p(w[0, |w| - 2], \sigma - 1)$, thus we can compute $p(w, \sigma)$ in $O(|w|(\sigma - |w|)^2)$ time using a dynamic programming algorithm [68]. Measuring the variance requires the probability that two overlapping windows host w as a subsequence at the same time. The authors of [68] take an enumerative, exponential approach, that explicitly iterates over all strings in Σ^σ that contain w as a subsequence. This algorithm has been generalized to arbitrary sets of patterns [20], to the set of all permutations of a given pattern, as well as to strings generated by variable-length Markov sources [69] and to multi-stream patterns with inter-stream dependencies of prescribed types [70]. In these richer setups, computing $p(w, \sigma)$ itself requires more complex recursions, and ingenious ways to traverse the corresponding recursion graphs have been devised.

Gapped patterns could also be represented as *regular expressions*. Computing the expected number and variance of the number of occurrences of a regular expression in an IID string Z typically requires converting the regular expression into a marked DFA, and then deriving the generating function of the language recognized by the automaton via the Chomsky-Schützenberger algorithm [117]. This pipeline can be extended to Markov chains, and editing the DFA allows to accommodate multiple languages as well as matches with errors [115]. The approach of using a DFA to compute statistics on regular expressions has also been used to compute the probability of a regular expression in a random string generated by IID and Markov sources (see e.g. [21] and references therein), and more recently it has enabled genome-scale compositional analyses of gapped patterns [116]. The conversion from a regular expression w to its

corresponding DFA, however, is exponential in $|w|$ in the worst case. A more restrictive way to model flexible gaps could be imposing $w = u \bullet_{d_1,d_2} v$, where $u \in U \subset \Sigma^+$, $v \in V \subset \Sigma^+$, and $\bullet_{d_1,d_2}$ is a flexible gap that extends for at least d_1 and at most d_2 positions. The probability that Z (assumed to be generated by a first-order Markov chain) contains *at least one occurrence* of w can be expressed as a function of random variables that relate to substrings, and whose probabilities can be computed recursively in $O((|u| + |v| + d_2)^2)$ time [141, 142, 143]. This approach holds for arbitrary sets U and V, and can be extended to strings with mismatches [143].

Discarding flexibility altogether, closed-form, fast approximations have been proposed for the expected number of distinct *rigid maximal motifs* with length ℓ, with k solid characters, and with exactly n occurrences in an IID string Z [161]. Intuitively, a rigid maximal motif (to be defined in Section 3.4) is a pattern with rigid gaps that occurs at least two times in Z, and that cannot be made more specific without losing support [122]. Most significance scores are monotonically nondecreasing with respect to motif specification, thus maximal motifs usually have the greatest significance among all motifs with the same support, and they embed any motif with exactly the same support and score (see e.g. [11] and references therein). Approximations like those in [161], or even simpler ones, back popular motif discovery tools (see e.g. [30, 137]), but crucially rely on the assumption that the occurrences of a motif w in Z are independent.

Independence is waived in [157], which gives exact formulas for the expectation and variance of the number of occurrences of a rigid gapped pattern with symbols in $\Gamma \subset 2^\Sigma$ in a random string generated by a Markov chain. These formulas will be detailed in Section 3.2; here we just remark that they are used at the core of a popular algorithm that discovers transcription factor binding sites in DNA [158, 159], and that the kernels of such computations iterate over a number of strings that grows exponentially in the length of the pattern, thus limiting this approach to very small

queries. In Section 3.3 we describe a simple observation that brings the running time of the formulas in [157] from $O(2^{|w|})$ to $O(|w|^2)$. In the IID case, the time to compute such formulas is dominated by convolution, and thus belongs to $O(|w| \log |w|)$. Given a string s provided offline, Section 3.4 studies the problem of computing expectation and variance for an arbitrary *motif* w of s provided *online*. We relate computational complexity to the structure of w and to the basis of *tiling motifs* of s, and we identify sets of rigid gapped motifs whose variance can be computed in less than $O(|w| \log |w|)$ time. The key idea behind our construction is reusing a suitable set of convolutions performed offline.

3.2 Notation and problem definition

In this section we summarize the algorithm described in [157], highlighting its computational kernels. For clarity of presentation, we adopt a slightly different notation. Let Σ be a finite alphabet, and let $\Gamma \subset 2^\Sigma \setminus \emptyset$ be such that $\{a\} \in \Gamma$ for all $a \in \Sigma$. We call *pattern* any string $s \in \Gamma^+$, and we say that a position i in s is a *gap* if $s[i] = \Sigma$. Given patterns s and t, we write $s \otimes_k t$ to mean a pattern of length $k + |t|$ with set $s[i] \cap t[i-k]$ at position i. We postulate $s[i] = \Sigma$ for $i \notin [0, |s| - 1]$, and we set $s \otimes_k t = \varepsilon$ if $s[i] \cap t[i-k] = \emptyset$ for some i. With $w \dashv s$ we indicate that pattern $w \in \Gamma^{|s|}$ is a copy of pattern s in which every nonsingleton character $G \in \Gamma$ that occurs in s has been transformed into a corresponding character $c \in G$: we call w an *instantiation* of s. In other words, an instantiation of pattern s forces all positions of s, except those occupied by a gap, to equal a symbol in Σ. With $w \prec s$ we indicate that *string* $w \in \Sigma^{|s|}$ is a copy of pattern s in which every character $G \in \Gamma$ occurring in s has been transformed into a corresponding character $c \in G$, including gaps.

Given a pattern s, we call *selector* a diagonal square matrix $\mathbf{I}(s, i)$ with $|\Sigma|^d$ rows, such that every diagonal element corresponding to a string $w \in \Sigma^d \ : \ w \prec s[i, i+d-1]$ is equal to one, and all other elements are zero. We overload the term *selector* to

include vectors as well: $\mathbf{e}(s,i)$ is a vector with $|\Sigma|^d$ components, such that every component corresponding to a string $w \in \Sigma^d \ : \ w \prec s[i, i+d-1]$ is one, and all other components are zero. Whether we will be referring to matrices or vectors will be clear from the context. Clearly $\mathbf{e}(s,i)$ (respectively, $\mathbf{e}(s,i)'$) is a right (respectively, left) eigenvector of $\mathbf{I}(s,i)$ associated with eigenvalue 1. For a pattern s, we set $\mathbf{u}(s,i)$ to be a vector with $|\Sigma|^d$ components, such that all components corresponding to strings $w \in \Sigma^d \ : \ w \prec s[i, i+d-1]$ are equal and sum to one, and all other components are zero.

Recall that our purpose is computing expectation and variance of the number of occurrences of a pattern in a random string. Consider thus a Markov chain of order d with matrix of transition probabilities $\mathbf{P} \in [0,1]^{|\Sigma|^d \times |\Sigma|^d}$ and stationary distribution $\mathbf{p} \in [0,1]^{|\Sigma|^d}$. We denote with t_{mm} the time to compute the product between two square matrices of size $|\Sigma|^d$, and with t_{vm} the time to compute the product between a vector of size $|\Sigma|^d$ and matrix of size $|\Sigma|^d$. Let $\mathbf{v} \in [0,1]^{|\Sigma|^d}$ be a vector of a priori probabilities for d-mers, let Z be a string generated by the Markov chain, and let s be a pattern with $|s| \leq |Z|$. With $p(s|\mathbf{v})$ we denote the probability that s occurs at position $0 \leq i \leq |Z| - |s|$ of Z in the form of one or more of its instantiations, assuming that $\mathbf{v}$ is the probability distribution of d-mers at i. If Z is long enough, it is safe to set $\mathbf{v} = \mathbf{p}$ independent of i (see e.g. [157] and references therein). Given a pattern w, we define the random variable X_w to be the number of occurrences of w in Z, and we set the indicator random variable $X_{w,i}$ to be one iff w occurs at position i in Z in the form of one or more of its instantiations. The expectation of X_s is clearly:

$$\mathbb{E}(X_s) \ = \ \sum_{w \in A(s)} (|Z| - |w| + 1) \cdot p(w|\mathbf{p}) \tag{2}$$

where $A(s) = \{w \dashv s\}$ and $p(w|\mathbf{p}) = \mathbf{p}' \cdot \mathbf{I}(w,0) \cdot \left[\prod_{i=1}^{|w|-d} \mathbf{P}\mathbf{I}(w,i)\right] \cdot \mathbf{e}(w, |w|-d)$. After standard manipulations, computing the variance of X_s reduces to Equation 2 and to the two following kernels, that relate to overlapping and nonoverlapping occurrences

of s, respectively:

$$\begin{aligned}
\sum_{i=0}^{|Z|-|s|-1}\sum_{j=i+1}^{|Z|-|s|}\mathbb{E}(X_{s,i}X_{s,j}) &= \sum_{i=0}^{|Z|-|s|-1}\sum_{l=1}^{m}\sum_{v\dashv s}\sum_{w\dashv s}\mathbb{E}(X_{v,i}X_{w,i+l}) \\
&= \sum_{w\in B(s)}(|Z|-|w|+1)\cdot p(w|\mathbf{p}) \qquad (3)\\
\sum_{i=0}^{|Z|-2|s|}\sum_{j=i+|s|}^{|Z|-|s|}\mathbb{E}(X_{s,i}X_{s,j}) &= \sum_{i=0}^{|Z|-2|s|}\sum_{l=0}^{|Z|-2|s|-i}\sum_{v\dashv s}\sum_{w\dashv s}\mathbb{E}(X_{v,i}X_{w,i+|s|+l}) \\
&= \sum_{w\in C(s)}(|Z|-|w|+1)\cdot p(w|\mathbf{p}) \qquad (4)
\end{aligned}$$

where $m=\min\{|s|-1,|Z|-i-|s|\}$, $B(s)=\{v\otimes_l w \mid v\dashv s, w\dashv s, 1\le l<|s|\}\setminus\{\varepsilon\}$ is the set of all valid overlaps of two instantiations of s, and $C(s)=\{v\otimes_{|s|+l} w \mid v\dashv s, w\dashv s, 0\le l\le |Z|-2|s|\}$ is the set of all spaced concatenation of two instantiations of s. Sets $A(s)$, $B(s)$ and $C(s)$ are enumerated explicitly in [157], thus computing Equations 2, 3 and 4 requires time, respectively:

$$\begin{aligned}
(|s|-d)\cdot(t_{vm}+|\Sigma|^d)\cdot|\Sigma|^{|s|} &\in O(|s|\cdot|\Sigma|^{|s|})\\
|\Sigma|^{2|s|}\cdot(t_{vm}+|\Sigma|^d)\cdot\sum_{l=1}^{|s|-1}(|s|+l-d) &\in O(|s|^2\cdot|\Sigma|^{|s|})\\
|\Sigma|^{2|s|}\cdot(t_{vm}+|\Sigma|^d)\cdot\sum_{l=1}^{|Z|-2|s|}(2|s|+l-d) &\in O(|Z|^2\cdot|\Sigma|^{|s|})
\end{aligned}$$

The dependence on $|Z|$ of Equation 4 is attenuated in [157] by introducing the following approximation[1]. For patterns v, w, we denote with $p(v \overset{l}{\mapsto} w)$ the probability of transitioning in exactly l steps of the Markov chain to any d-mer $y \prec w[0,d-1]$, starting from any d-mer $x \prec v[|v|-d,|v|-1]$. Formally, $p(v \overset{l}{\mapsto} w) = \mathbf{u}(v,|v|-d)'\cdot\mathbf{P}^l\cdot\mathbf{e}(w,0)$. For a pattern w, we similarly denote with $p(\mapsto w)$ the probability that w occurs, assuming that any d-mer $x\prec w[0,d-1]$ occurs. Formally, $p(\mapsto w) =$

[1]Alternatively, we could shave a $O(|Z|)$ factor from the running time of Equation 4 by using the expression for sums of powers of stochastic matrices given in [86]. We use this method in Section 3.3.

$\mathbf{u}(w,0)' \cdot \left(\prod_{i=1}^{|w|-d} \mathbf{PI}(w,i)\right) \cdot \mathbf{e}(w,|w|-d)$. Equation 4 can thus be approximated by:

$$\sum_{v \dashv s} \sum_{w \dashv s} p(v|\mathbf{p}) \cdot p(\mapsto w) \cdot \sum_{l=1}^{|Z|-2|s|} (|Z|-2|s|-l+1) \cdot p(v \overset{l+d}{\longmapsto} w)$$

and can thus be computed in time $O(|Z| \cdot |\Sigma|^{|s|})$.

The exponential dependency of running time on $|s|$ is not a problem in the specific application domain of [157], where pattern have length approximately 20 and the alphabet has size 4, but it makes it impractical to generalize this approach to patterns of arbitrary length. In Section 3.3 we describe a way to make the computation of Equations 2, 3 and 4 scale quadratically, rather than exponentially, on $|s|$.

3.3 Gapped patterns

Equations 2, 3 and 4 can be computed without explicitly iterating over sets $A(s)$, $B(s)$ and $C(s)$. Avoiding the explicit construction of such sets brings both an asymptotic speedup, and the practical advantage of removing string operations altogether from the implementation of the corresponding equations.

Lemma 6. *Let s be a pattern, and let $\mathbf{v}$ be a vector of d-mer probabilities. Then, $p(s|\mathbf{v}) = \mathbf{v}'\mathbf{I}(s,0) \cdot \mathbf{PI}(s,1) \cdot \mathbf{PI}(s,2) \cdots \mathbf{Pe}(s,|s|-d)$.*

Proof. Clearly $\mathbf{I}(s,i) = \sum_{w \dashv s[i,i+d-1]} \mathbf{I}(w,0)$, and the matrices in this sum select disjoint subsets of Σ^d. Similarly, $\mathbf{e}(s,i) = \sum_{w \dashv s[i,i+d-1]} \mathbf{e}(w,0)$, and the vectors in the sum select disjoint subsets of Σ^d. Thus, $\mathbf{v}'\mathbf{I}(s,0) \cdot \mathbf{PI}(s,1) \cdot \mathbf{PI}(s,2) \cdots \mathbf{Pe}(s,|s|-d)$ can be written as:

$$\begin{aligned} & \mathbf{v}' \left(\sum_{w \dashv s[0,d-1]} \mathbf{I}(w,0) \right) \cdot \mathbf{P} \left(\sum_{w \dashv s[1,d]} \mathbf{I}(w,0) \right) \cdots \mathbf{P} \cdot \left(\sum_{\substack{w \dashv s[|s|-d,\\ |s|-1]}} \mathbf{e}(w,0) \right) \\ = & \sum_{\substack{w_0 \dashv s[0,d-1],\\ \vdots \\ w_{|s|-1} \dashv s[|s|-d,|s|-1]}} \mathbf{v}'\mathbf{I}(w_0,0) \cdot \mathbf{PI}(w_1,0) \cdots \mathbf{P} \cdot \mathbf{e}(w_{|s|-d},0) \end{aligned}$$

Two selectors $\mathbf{I}(w_i, 0)$ and $\mathbf{I}(w_j, 0)$ in the sum above are called *incompatible* if either $i < j < i + d$ and $w_i \otimes_{j-i} w_j = \varepsilon$, or $j < i < j + d$ and $w_j \otimes_{i-j} w_i = \varepsilon$. By the structure of $\mathbf{P}$, products containing incompatible selectors do not contribute to the sum, and set $\{(w_0, w_1, \ldots, w_{|s|-d}) \mid \nexists\ 0 \leq i, j \leq |s| - d : w_i, w_j \text{ are incompatible}\}$ can be put in one-to-one correspondence with $A(s)$. □

Equation 2 thus reduces to:

$$(|Z| - |s| + 1) \cdot \mathbf{p}'\mathbf{I}(s, 0) \cdot \mathbf{PI}(s, 1) \cdot \mathbf{PI}(s, 2) \cdots \mathbf{Pe}(s, |s| - d) \tag{5}$$

which can be computed in $O(|s|)$ time. Applying Lemma 6 to Equation 4 we get:

$$\mathbf{p}'\mathbf{I}(s,0)\cdot\mathbf{PI}(s,1)\cdots\mathbf{PI}(s,|s|-d)\cdot\left(\sum_{i=0}^{|Z|-2|s|}\sum_{l=0}^{|Z|-2|s|-i}\mathbf{P}^{l+d}\right)\mathbf{I}(s,0)\cdot\mathbf{PI}(s,1)\cdots\mathbf{Pe}(s,|s|-d)$$

After using the expression for sums of powers of stochastic matrices given in [86], this becomes:

$$\begin{aligned}\mathbf{q}'\mathbf{Q}\mathbf{P}^{n-2|s|+3}\mathbf{Q}\mathbf{P}^{d-1}\mathbf{r} - \mathbf{q}'\mathbf{Q}\mathbf{P}^2\mathbf{Q}\mathbf{P}^{d-1}\mathbf{r} + (n-2|s|+1)\mathbf{q}'\mathbf{Q}\mathbf{1}\mathbf{p}'\mathbf{P}^{d-1}\mathbf{r} + \\ -(n-2|s|+1)\mathbf{q}'\mathbf{Q}\mathbf{P}^d\mathbf{r} + \left(\frac{n^2}{2} + 2|s|^2 - 2|s|n + 1 + \frac{3}{2}n - 3|s|\right)\mathbf{q}'\mathbf{1}\mathbf{p}'\mathbf{P}^{d-1}\mathbf{r}\end{aligned} \tag{6}$$

where $\mathbf{q}' = \mathbf{p}'\mathbf{I}(s, 0) \cdot \mathbf{PI}(s, 1) \cdots \mathbf{PI}(s, |s| - d)$, $\mathbf{r} = \mathbf{I}(s, 0) \cdot \mathbf{PI}(s, 1) \cdots \mathbf{Pe}(s, |s| - d)$, $\mathbf{1}$ is the vector of $|\Sigma|^d$ ones, and $\mathbf{Q} = (\mathbf{P} - \mathbf{I} + \mathbf{1}\mathbf{p}')^{-1}$ as defined in [86]. This equation can be computed in $O(|Z|)$ time as is, or in constant time if we assume $|Z| \gg |s|$ and thus approximate $\mathbf{P}^{|Z|-2|s|+3}$ with $\mathbf{1}\mathbf{p}'$ as done in [157]. Equation 3 can similarly be computed in $O(|s|^2)$ time:

$$\sum_{\substack{w = s \otimes_k s, \\ 1 \leq k < |s|}} (|Z| - |s| - k + 1) \cdot \mathbf{p}'\mathbf{I}(w, 0) \cdot \mathbf{PI}(w, 1) \cdot \mathbf{PI}(w, 2) \cdots \mathbf{Pe}(w, |w| - d) \tag{7}$$

Lemma 6 can speed up also the parts of [157] that depend on the specific domain of transcription factor binding sites. Assume that $\Sigma = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$ and let the *complement* of a $\mathtt{a}$ be $\mathtt{t}$ and the complement of $\mathtt{c}$ be $\mathtt{g}$, and vice versa. Complementation

extends naturally to subsets of Σ. Denote with $\bar{s}$ the *reverse complement* of a pattern s. The probability that s occurs at a generic position of a random string Z, possibly in the form of $\bar{s}$, is just $p'(s|\mathbf{v}) = p(s|\mathbf{v}) + p(\bar{s}|\mathbf{v})$ (any instantiation $v \in \Gamma^{|s|}$ of s such that $v \dashv s$ and $v \dashv \bar{s}$ is counted twice in [157], thus no further correction is needed). Adapting Equations 5, 6 and 7 is thus straightforward. When the reverse complement of s is considered, however, a fourth term needs to be taken into account in the variance:

$$\sum_{i=0}^{|Z|-|s|} \sum_{v \dashv s} \mathbb{E}\left(\sum_{w \dashv \bar{s}} X_{v,i} X_{w,i}\right) = \sum_{w \in D(s)} (|Z| - |w| + 1) \cdot p(w|\mathbf{p})$$

where $D(s) = \{v \otimes_0 w \mid v \dashv s, w \dashv \bar{s}\} \setminus \{\varepsilon\}$. This quantity is clearly $(|Z| - |s| + 1) \cdot p(s \otimes_0 \bar{s}|\mathbf{p})$.

Let's now return to Equation 7. Clearly $s \otimes_k s \neq \varepsilon$ iff $|s| - k$ is a *border* of s, i.e. if the prefix of length $|s| - k$ of s matches the suffix of length $|s| - k$ of s. For strings in Σ^+, borders have a recursive structure that enables the enumeration of all borders of a string in overall linear time. As mentioned in the introduction, a version of Equation 7 for the IID case can be embedded in the computation of borders, thereby requiring just $O(|s|)$ time to compute [8]. This scheme can be easily extended to Markov chains of any order if we remain in Σ^+, but it breaks down for strings in Γ^+ [78]. When $s \in \Gamma^+$ and Z is generated by an IID source, Equation 7 becomes:

$$\sum_{\substack{w = s \otimes_{|s|-b} s, \\ b \in \beta(s)}} (|Z| - 2|s| + b + 1) \cdot \left(\prod_{i=0}^{|s|-b-1} \mathbb{P}(s[i])\right) \cdot \left(\prod_{i=|s|-b}^{|s|-1} \mathbb{P}(w[i])\right) \cdot \left(\prod_{i=b}^{|s|-1} \mathbb{P}(s[i])\right) \quad (8)$$

where the first and third products can be accessed in constant time after $O(|s|)$ preprocessing of s. The second term is clearly convolutional, and it can be computed in $O(|s| \log |s|)$ time using the landmark match-count algorithm for string searching by Fischer and Paterson [57], or one of its recent, randomized variants (see e.g. [38]). In the next section, we study ways to bring the complexity of this computation below $O(|s| \log |s|)$ in the case in which patterns are queried online, but we know that they

are *motifs* of a specific string provided offline.

3.4 Motifs

In most applications we are given a fixed text $s \in \Sigma^+$ and a *set* of patterns w_0, w_1, ..., w_{k-1} in Γ^+. If patterns are provided offline, the fastest way to compute expectation and variance with respect to a random Markovian string would be to apply Equations 5, 6 and 7 to each pattern separately. Storing patterns in a suffix tree would allow to reuse some intermediate results in practice, without however affecting asymptotics (see e.g. [69]). Assume, on the other hand, that string s is given offline, and that we are asked to compute the expectation and variance of *arbitrary* patterns provided *online*. This scenario models popular websites that allow to search for biologically significant patterns in genomes and proteomes (e.g. [155]), and captures the post-processing stage of most pattern-discovery algorithms, which rank their results according to statistical significance (e.g. [30]). In what follows we will focus on the IID case and on computing Equation 8. The main intuition behind performing less than $O(|w| \log |w|)$ operations for a pattern w given online is moving some convolutions offline, and reusing such convolutions at query time with the help of suitable data structures.

Given a string $w \in \Gamma^+$, we define $\sharp_{w,a}[i]$ to be the number of positions in which $w \otimes_i w$ equals $a \in \Gamma$, $0 \leq i < |w|$. We define $\sharp_{w,a}[i,j,k]$ to be the number of positions in which $w[i, i+k-1] \otimes_0 w[j, j+k-1]$ equals $a \in \Gamma$, $0 \leq i, j \leq |w|-k$. Finally, we define $\sharp_{s,t,a}[i]$ to be the number of positions in which $s \otimes_i t$ equals $a \in \Gamma$, $-|t|+1 \leq i < |s|$. To simplify notation, we use symbol $\bullet$ to denote set Σ, and we indicate with $||w||$ the number of positions in which w is different from $\bullet$. We first study ways in which the convolution of w can be reused to compute the convolution of its prefixes and suffixes.

Lemma 7. *Let $w \in \Sigma(\Sigma \cup \{\bullet\})^*\Sigma$, and assume that $\sharp_{w,a}[i]$ is known for every $1 \leq$*

$i < |w|$ *and every* $a \in \Sigma$. *Then, the value of* $\sharp_{w_k,a}[i]$ *for* $1 \leq i \leq k$ *and* $a \in \Sigma$ *can be computed for all prefixes* $w_k = w[0,k]$ *of* w *(similarly, the value of* $\sharp_{w_k,a}[i]$ *for* $1 \leq i < |w| - k$ *and* $a \in \Sigma$ *can be computed for all suffixes* $w_k = w[k, |w| - 1]$ *of* w*) in overall optimal* $O(|w|^2)$ *time and space and in overall* $O(||w|| \cdot |w|)$ *arithmetic operations.*

Proof. For a generic offset i, $\sharp_{w[0,|w|-2],a}[i] = \sharp_{w,a}[i] + \tau_{a,i,|w|-2}$, where:

$$\tau_{a,i,j} = -\sharp_{w,a}[j+1, j+1, 1] - \sharp_{w,a}[j+1, j-i, 1] + \sharp_{w,a}[j-i, j-i, 1]$$

In particular, $w[j+1] = \bullet$ implies $\tau_{a,i,j} = 0$ for all i and $a \in \Sigma$. Let $\mathbf{T}_a$ be an upper-triangular matrix with $|w|-2$ rows and columns, indexed starting from one, in which row i corresponds to offset i, column j corresponds to prefix $w[0,j]$, and $\mathbf{T}_a[i,j] = \tau_{a,i,j}$. Matrix $\mathbf{T}_a$ is filled in column-major order, starting from column $|w| - 2$. It is easy to see that $\mathbf{T}_a$ has a regular structure (Figure 40a). First, as mentioned above, since $w[|w|-1] \in \Sigma$, the fact that $\mathbf{T}_a[i, |w|-2] = -2$ implies that $\mathbf{T}[k, |w|-i-2] = 0$ for all $1 \leq k \leq |w| - i - 2$. Second, let $j_a = \max\{j : w[j] = a, 0 \leq j < |w|\}$; then, every column $\mathbf{T}_a[:, j]$ such that $w[j+1] = a$ equals column $\mathbf{T}[:, j_a - 1]$ shifted up by $j_a - j - 1$ cells. Third, there is only one other type of column in $\mathbf{T}_a$, not considering shifts and columns that are identically zero: the column corresponding to symbols different from a, which contains only zeros and ones, and appears sequentially shifted up (for $j < j_a - 1$) and down (for $j > j_a - 1$) as described above. We can thus compute these two types of column in $O(|w|)$ time and space, and store them rather than $\mathbf{T}_a$ itself in practice. To compute $\sharp_{w[0,k],a}$ for every k, traverse the columns of $\mathbf{T}_a$ from right to left, keeping a running sum of the cells associated with every row i. If $\mathbf{T}_a[i,j] = 0$, then the corresponding $\sharp_{w[0,j],a}[i]$ is just copied from $\sharp_{w[0,j+1],a}[i]$. □

Corollary 1. *Let* $w \in \Sigma(\Sigma \cup \{\bullet\})^*\Sigma$*, and let* $\mathbf{X}_w$ *be a vector with* $|w|$ *components, such that:*

$$\mathbf{X}_w[i] = (|Z| - |w| - i + 1) \cdot \prod_{j=0}^{i-1} \mathbb{P}(w[j]) \cdot \prod_{j=i}^{|w|-1} \mathbb{P}(v_i[j]) \cdot \prod_{j=|w|-i}^{|w|-1} \mathbb{P}(w[j])$$

where $v_i = w \otimes_i w$ *and* $\mathbb{P}(\emptyset) = 1$. *Assume that we know* $\mathbf{X}_w[i]$ *for all* i. *Then, we can compute Equation 8 for all prefixes and all suffixes of* w *in overall* $O(||w|| \cdot |w|)$ *time and optimal* $O(|w|)$ *space.*

Proof. We follow closely the proof of Lemma 7. For a generic offset i:

$$\mathbf{X}_{w[0,|w|-2]}[i] \;=\; \mathbf{X}_w[i] \cdot \tau_{i,|w|-2} \cdot \left(1 + \frac{1}{|Z| - |w| - i + 1}\right) \cdot \frac{1}{\mathbb{P}(w[|w|-1])}$$

where $\tau_{i,j} = \mathbb{P}(w[j-i+1])/\mathbb{P}(w[j+1] \cap w[j-i+1])$ if $w[j+1]$ matches $w[j-i+1]$, and $\tau_{i,j} = \mathbb{P}(w[j-i+1])$ otherwise. Clearly $w[j+1] = \bullet$ implies $\tau_{i,j} = 1$ for all i. We are interested in the case $|Z| \gg 2|w|$, thus the second factor in the above equation could be considered independent of i in practice. $\tau_{i,j}$, however, does depend on i. Let $\mathbf{T}$ be an upper-triangular matrix with $|w| - 2$ rows and columns, indexed starting from one, in which row i corresponds to offset i, column j corresponds to prefix $w[0, j]$, and $\mathbf{T}[i, j] = \tau_{i,j}$. Matrix $\mathbf{T}$ is filled in column-major order, starting from column $|w| - 2$, and it has, again, a regular structure (Figure 40b). First, as mentioned above, since $w[|w|-1] \in \Sigma$, the fact that $\mathbf{T}[i, |w|-2] = 1/\mathbb{P}(w[|w|-1])$ implies that $\mathbf{T}[k, |w|-i-2] = 1$ for all $1 \leq k \leq |w|-i-2$. Second, let $j_a = \max\{j : w[j] = a, 0 \leq j < |w|\}$; then, every column $\mathbf{T}[:, j]$ such that $w[j+1] = a$ equals column $\mathbf{T}[:, j_a - 1]$ shifted up by $j_a - j - 1$ cells. Third, there are just $|\Sigma|$ distinct types of columns in $\mathbf{T}$, not considering shifts and null columns. Indeed, let $w[|w|-1] = a$; for every j such that $w[j] = b \neq a$, $b \in \Sigma$, we have that $\mathbf{T}[:, j-1]$ is a copy of $\mathbf{T}[:, |w|-2]$ shifted up by $|w| - j - 1$ cells, where every one is replaced by a, every b is replaced by one, and every $1/\mathbb{P}(a)$ is replaced by $1/\mathbb{P}(b)$. We can thus compute and store these $|\Sigma|$ types of columns in $O(|w|)$ time and space overall, rather than storing $\mathbf{T}$ itself. To compute Equation 8 for all prefixes of w, traverse the columns of $\mathbf{T}$ from right to left, keeping a running product of the cells associated with every row i, and keeping a count of the number of matches associated with every row i as done in Lemma 7. Then, sum the values of every column corresponding to rows with no mismatches. The value of

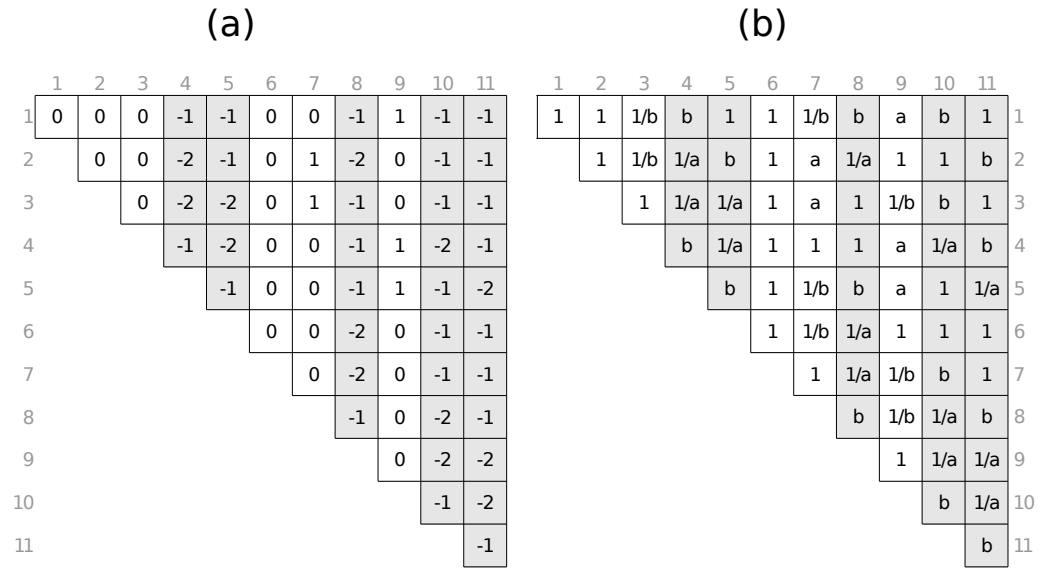

Figure 40: Matrix $\mathbf{T}_a$ of Lemma 7 (a) and matrix $\mathbf{T}$ of Corollary 1 (b) for string $ab \bullet \bullet baa \bullet babaa$. To avoid clutter, $\mathbb{P}(a)$ is abbreviated with a for all symbols $a \in \Sigma$. Light gray highlights column 11 and its shifts.

Equation 8 associated with a column that is entirely one needs just constant time to be derived from the value of the previous column. □

Lemma 7 and Corollary 1 generalize easily to suffixes and to strings in Γ^+. Given a string w, in what follows we will call $\mathcal{P}_{w,a}$ the upper-triangular matrix with $|w| - 2$ rows and columns, indexed starting from one, in which row i corresponds to offset i, column j corresponds to prefix $w[0, j]$, and $\mathcal{P}_{w,a}[i, j] = \sharp_{w[0,j],a}[i]$. We similarly define $\mathcal{S}_{w,a}$ for suffixes. A nice property of $\mathcal{P}_{w,a}$ and $\mathcal{S}_{w,a}$ is that they can be used as indexes to answer questions about arbitrary substrings of w in constant time[2].

Lemma 8. *Let $w \in \Gamma^+$. After $O(|w|^2)$ preprocessing, we can compute the following quantities: (1) $\sharp_{v,a}[k]$ for all $1 \leq k < |v|$ and $a \in \Gamma$ in $O(|v|)$ time, for any substring v of w; (2) $\sharp_{w_1,w_2,a}[i]$ for all $-|w_2| + 1 \leq i < |w_1|$ and $a \in \Gamma$ in $O(|w_1| + |w_2|)$ time, for any pair of substrings w_1 and w_2 of w.*

Proof. (1) Build $\mathcal{P}_{w,a}$ in $O(|w|^2)$ time. Then, build the suffix tree $\mathcal{T}_w$ of w, and assign

[2]From this point of view, such data structures recall the *match matrix* described in [17].

to every internal node the starting position of one of the suffixes in its subtree. This can be done in overall $O(|w|)$ time. Given a substring v of w, find its proper locus in $\mathcal{T}_w$, extract the associated starting position i in w, set $j = i + |w| - 1$, and apply the following identity to every $a \in \Gamma$ (see Figure 41):

$$\sharp_{w[i,i+|w|-1],a}[k] = \sharp_{w[0,i+|w|-1],a}[k] - \sharp_{w[0,i+k-1],a}[k] + 2 \cdot \sharp_{w,a}[i,i,k] \tag{9}$$

The first two terms in the right-hand side can be computed from $\mathcal{P}_{w,a}$, and the last term can be accessed in constant time after $O(|w|)$ preprocessing of w. The equation above could be set up for using $\mathcal{S}_{w,a}$ rather than $\mathcal{P}_{w,a}$. Notably, just one of $\mathcal{P}_{w,a}$ and $\mathcal{S}_{w,a}$ suffices to answer queries on single substrings. (2) Preprocess w as above. Let i_1 and i_2 be the starting positions of w_1 and w_2 in w, respectively. For clarity of presentation, we describe the formula for the case in Figure 41, leaving the general case to the reader. Let $k' = i_2 - i_1 - k$ and $i^* = i_1 + |w_1|$. Then:

$$\begin{aligned}
\sharp_{w_1,w_2,a}[k] \;=\; & \sharp_{w,a}[k'] + \\
& -\sharp_{w[0,i_2-1],a}[k'] + \sharp_{w,a}[i_1+k, i_1+k, k'] + \sharp_{w,a}[i_1,i_1,k] + && (10)\\
& -\sharp_{w[i_1+|w_1|,|w|-1],a}[k'] + \sharp_{w,a}[i_1+|w_1|, i_1+|w_1|, k'] + && (11)\\
& +\sharp_{w,a}[i^*+k', i^*+k', i_2+|w_2|-i^*-k'] && (12)
\end{aligned}$$

where all terms can be accessed in constant time by either querying $\mathcal{P}_{w,a}$ and $\mathcal{S}_{w,a}$, or by accessing values that have been precomputed in $O(|w|)$ time. □

In what follows, we will also be interested in computing $\sharp_{v,a}[i]$ for a string v that is *less specific* than a known string w.

Definition 16. *Given a string $w \in \Gamma^+$, we say that string $v \in \Gamma^{|w|}$ is* less specific *than w (or, equivalently, that v is a* sparsification *of w) if $w[i] \subseteq v[i]$ for all $0 \leq i < |w|$, and if $w[i^*] \subset v[i^*]$ in at least one position i^*.*

When the maximum distance between two sparsified positions of v is bounded by a sublinear function of $|w|$, computing the convolution of v by exploiting the

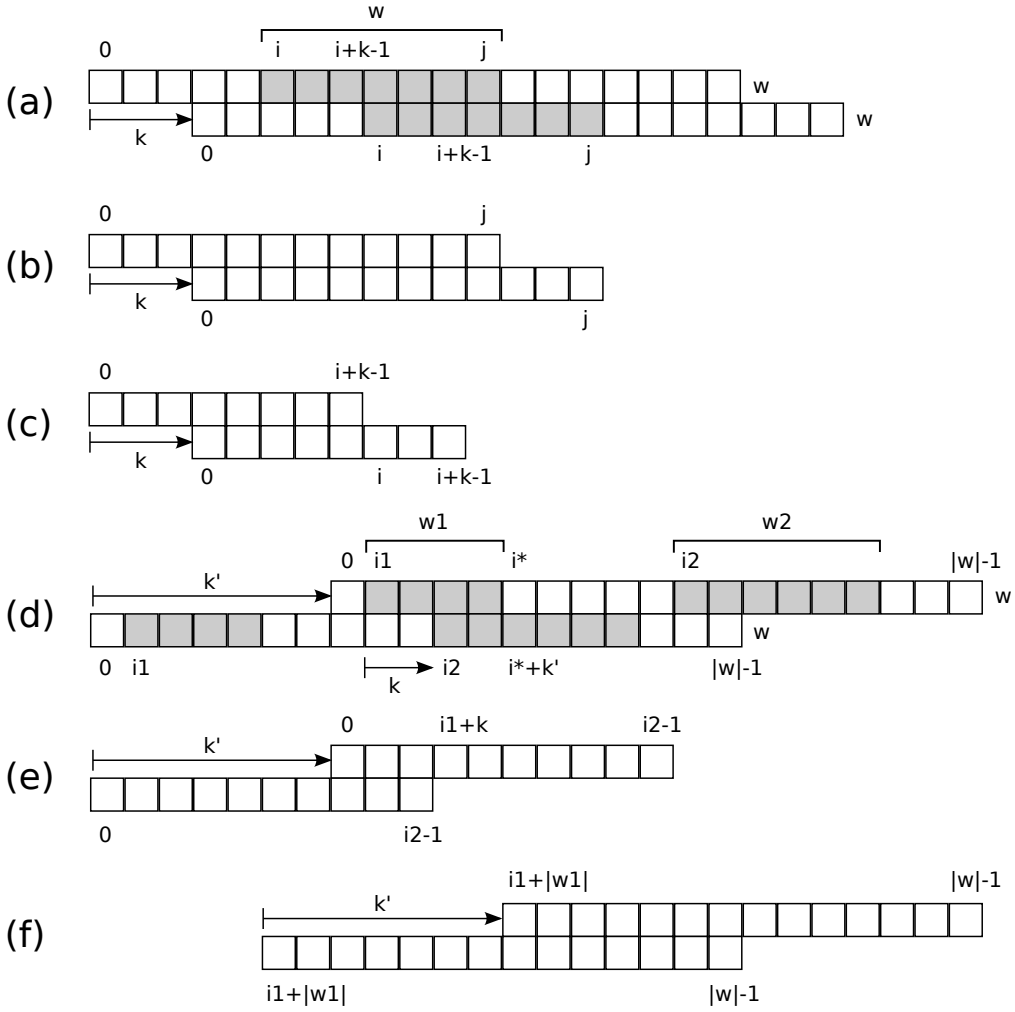

Figure 41: Illustrating Lemma 8. (a) Reusing the convolution of a string w to compute the convolution of a substring. (b,c) The prefixes of w used in Equation 9. (d) Reusing the convolution of a string w to compute the convolution between two substrings w_1 and w_2 of w. (e) The prefix of w used in Expression 10. (f) The suffix of w used in Expressions 11 and 12.

convolution of w is asymptotically faster than computing the convolution of v with no prior information.

Lemma 9. *Let w be a string in $(\Sigma \cup \{\bullet\})^+$, and let $v \in (\Sigma \cup \{\bullet\})^{|w|}$ be a sparsification of w such that $v[i] = w[i]$ for all $0 \leq i < |w|$, except for a set of positions $V = \{i_0, i_1, \ldots, i_{k-1}\}$ where $w[i_j] \subset v[i]$, $0 \leq j < k$. We can compute $\sharp_{v,a}[i]$ from $\sharp_{w,a}[i]$ for all offsets i in overall $O(|v| \log(i_{k-1} - i_0))$ time.*

Proof. For every $a \in \Sigma$, let $\{i_0^a, i_1^a, \ldots, i_{k_a-1}^a\}$ be the partition of V in which $w[i_j^a] = a$ for $0 \leq j < k_a$. Project substring $v[i_0^a, i_0^a + 1, \ldots, i_{k_a-1}^a]$ to a binary vector $\mathbf{V_a}$ of length $i_{k_a-1}^a - i_0^a + 1$ that is zero everywhere, except at positions $i_j^a - i_0^a$, at which it equals one. Then, compute in $O((i_{k_a-1}^a - i_0^a) \log(i_{k_a-1}^a - i_0^a))$ time the convolution of $\mathbf{V_a}$ with itself, and store the result in vector $\mathbf{X_a}$: thus, $\mathbf{X_a}[i]$ contains the number of new gaps in v that are aligned with other new gaps in v in $v \otimes_i v$. Similarly, project w to a binary vector $\mathbf{W}$ of length $|w| + i_{k_a-1}^a - i_0^a + 1$ that is zero everywhere, except at positions of w that contain a gap, and at positions greater than or equal to $|w|$. Compute in $O(|v| \log(i_{k_a-1}^a - i_0^a))$ time the convolution of $\mathbf{W}$ and $\mathbf{V_a}$, storing the result in vector $\mathbf{Y_a}$: thus, $\mathbf{Y_a}[i]$ stores the number of times in $w \otimes_i v[i_0^a, \ldots, i_{k_a-1}^a]$ a new gap in v is aligned with a gap that already existed in w. Any offset i that is valid for w is also valid for v, and the number of matching characters can be corrected in constant time as follows:

$$\sharp_{v,a}[i] = \sharp_{w,a}[i] - \mathbf{X_a}[i] - \mathbf{Y_a}[i_0^a + i] - \mathbf{Y_a}[i_0^a - i]$$

An offset i that is *not* valid for w, on the other hand, could become valid for v. Convolve vector $\mathbf{V_a}$ with the binary vector $\mathbf{V_b}$ of length $|v| + i_{k_a-1}^a - i_0^a + 1$, that is one only at positions i where $v[i] = b \neq a$. This convolution takes $O(|v| \log(i_{k_a-1}^a - i_0^a))$ time, and produces a vector $\mathbf{Z_a}$ that contains, at coordinate $\mathbf{Z_a}[i]$, the number of (a, b)-mismatches in $w \otimes_{i-i_0^a} w$ that can be removed by putting gaps in w at positions $\{i_0^a, i_1^a, \ldots, i_{k_a-1}^a\}$. Since we know the original number of matches in $w \otimes_i w$ for every

symbol in Σ and every invalid offset i, we can compute $\sharp_{v,a}[i]$ for every a and invalid offset i in constant time. □

Lemma 9 can be easily generalized to strings in Γ^+, and can be extended to *pairs* of strings w_1, w_2 with different sparsifications, at cost $O((|w_1|+|w_2|)\log(|w_1|+|w_2|))$. Another natural way to constrain the sparsification of w is forcing sparsified positions to occur inside a contiguous interval.

Lemma 10. *Let w be a string in $(\Sigma\cup\{\bullet\})^+$, and let $v \in (\Sigma\cup\{\bullet\})^{|w|}$ be a sparsification of w such that $v[i] = w[i]$ for all $0 \le i < |w|$, except possibly at positions $V = \{d, d+1, \ldots, d+k-1\}$ where $v[d+i] = \bullet$, $0 \le i < k$. After $O(||w||\cdot|w|)$ preprocessing, we can compute $\sharp_{v,a}[i]$ for all i in $O(|v|)$ time.*

Proof. Let $\overrightarrow{\sharp}_{w,a}[x, y, z]$ be the number of positions in $w[x, x+z-1] \otimes_0 w[y, y+z-1]$ that have an a in $w[x, x+z-1]$ and a gap in $w[y, y+z-1]$. Such values can be computed for w using convolution, then they can be propagated to all prefixes and suffixes of w following a strategy similar to the proof of Lemma 7. The resulting matrices support substring queries as described in Lemma 8, thus enabling the computation of the following correction in constant time:

$$\sharp_{v,a}[i] = \sharp_{w,a}[i] - \sharp_{w,a}[d+i, d, k-i] - \overrightarrow{\sharp}_{w,a}[d+k-i, d+k, i] - \overrightarrow{\sharp}_{w,a}[d, d-i, i]$$

□

Lemma 10 can be seen as applying to subsequences of w whose elapsing positions are gaps in w, and it can be easily generalized to handle pairs of strings with different sparsifications, as well as strings in Γ^+. In a practical implementation, Lemma 9 and Lemma 10 are applied in cascade to a string w. Since there are multiple ways in which w could be parsed into the tokens that are input to such lemmas, it is natural to ask for the fastest configuration.

Lemma 11. *Let $w \in (\Sigma \cup \{\bullet\})^+$, and let v be a sparsification of w. The set of applications of Lemmas 9 and 10 that leads to the fastest computation of $\sharp_{v,a}[i]$ for all $1 \leq i < |w|$ and $a \in \Sigma$ can be determined in $O(|w|^3)$ time.*

Proof. Both operations work on nonoverlapping substrings of w, thus let $G = (V, \overrightarrow{E})$ be a directed graph in which $V = \{w[i,j] \ : \ 0 \leq i \leq j < |w|\} \cup \{\alpha, \Omega\}$, where α and Ω are artificial nodes. Set $(w[i_1, j_1], w[i_2, j_2]) \in \overrightarrow{E}$ iff $i_2 > j_1$ and $(v[k] = \bullet) \Rightarrow (w[k] = \bullet)$ for $j_1 + 1 \leq k < i_2$. Set $(\alpha, w[i^*, j]) \in \overrightarrow{E}$ for all j, where $i^* = \min\{i : v[i] = \bullet \wedge w[i] \neq \bullet\}$. Finally, set $(w[i, j^*], \Omega) \in \overrightarrow{E}$ for all i, where $j^* = \max\{j : v[j] = \bullet \wedge w[j] \neq \bullet\}$. Assign to each vertex a cost: α, Ω, and all substrings of w that are not sparsified in v have cost zero. The vertex associated with a sparsified substring $w[i,j]$ has the cost of applying Lemma 10 to $w[i,j]$ if $v[i,j] \in \bullet^+$, and it has the cost of applying Lemma 9 to $w[i,j]$ otherwise. The set of operations that leads to the fastest computation of $\sharp_{v,a}[i]$ is the (α, Ω) path with smallest cost in G, and can thus be computed in $O(|w|^3 + |w|^2 \log |w|)$ time using Dijkstra's algorithm as implemented in [60]. □

Lemma 11 can be useful in a scenario in which w is fixed and sparsifications are queried online. Assuming that the structure of such sparsifications belongs to few known types, we could compute the best sparsification scheme for each type offline, and then just apply it online. Lemma 11 is particularly practical because it can use the running time of concrete implementations of Lemmas 9 and 10.

Recall that our purpose is preprocessing a given text $s \in \Sigma^+$ to compute the expectation and variance of arbitrary patterns $w \in \Gamma^+$ provided *online*. From now on, we will restrict to a specific class of patterns, called *motifs*.

Definition 17 (Motif [122])**.** *Given a string $s \in \Sigma^+$, a* motif *is a string $w \in \Sigma(\Sigma \cup \{\bullet\})^*\Sigma$ that occurs at least two times in s.*

The number of distinct motifs in a string s grows exponentially with $|s|$. Among the set of all motifs of s, a notable subset cannot be intuitively made "more specific"

without losing support.

Definition 18 (Maximal motif [122])**.** *Let w be a motif occurring at positions $\mathcal{L}(w) = \{i_0, i_1, \ldots, i_{n-1}\}$ in a string $s \in \Sigma^+$, $n \geq 2$. We say that w is* maximal in composition *if no other motif $v \neq w$ of s has $\mathcal{L}(v) = \mathcal{L}(w)$ and $v[i] \subseteq w[i]$ for all $i \in \{0, \ldots, |w| - 1\}$. We say that w is* maximal in length *if no other motif $v \neq w$ of s is such that $|\mathcal{L}(v)| = |\mathcal{L}(w)|$ and w is a substring of v. We say that w is a* maximal motif *of s if it is both maximal in composition and maximal in length.*

Unfortunately, even the number of maximal motifs can grow exponentially in $|s|$. A landmark result in pattern discovery states that the subset of *tiling* maximal motifs is bounded by a linear function of $|s|$ [122, 126].

Definition 19 (Tiling motif [126])**.** *A maximal motif w of a string s is* tiled *is there exist maximal motifs $w_0, w_1, \ldots, w_{n-1}$ of s ($w_i \neq w \; \forall \; i$) and integers $d_0, d_1, \ldots, d_{n-1}$ such that $\mathcal{L}_s(w) = \bigcup_{i=0}^{n-1} \mathcal{L}_s(w_i) + d_i$. We call* tiling *a maximal motif of s that is not tiled.*

The set of tiling motifs of s, together with their occurrence lists, contains sufficient information to generate any other maximal motif in s and its occurrences, without knowing s itself [122, 125]. It is thus standard to call this set a *basis*: in what follows, we will denote it with $\mathcal{B}_s$. We are interested here in the mechanism by which the basis generates a motif of s.

Fact 2 ([127])**.** *The motifs of s are all and only the strings in $\Sigma(\Sigma \cup \{\bullet\})^*\Sigma$ that can be obtained as follows: (1) take a substring of a tiling motif that starts and ends with a character in Σ; (2) replace an arbitrary set of solid characters (excluding the first and last ones) with gaps.*

This fact, combined with the sparsification tools described above, will be the core of our construction. Before describing the main result of this section, however, we need some more notation.

Definition 20. *Let $s \in \Sigma^+$ be a string, and let $\mathcal{B}_s$ be its tiling basis. We say that a string $w \in (\Sigma \cup \{\bullet\})^+$ is* gap-maximal *if it is a right-maximal substring of $\mathcal{B}_s$ such that $w\bullet$ does not occur in $\mathcal{B}_s$.*

Definition 21. *The* gap-factorization *of a string $w \in (\Sigma \cup \{\bullet\})^+$ induced by a string $s \in \Sigma^+$ is the decomposition $w = u_0 \bullet^{d_0} u_1 \bullet^{d_1} \cdots \bullet^{d_{k-2}} u_{k-1}$, where each u_i starts with a solid character, and is the longest prefix of $u_i u_{i+1} \dots u_{k-1}$ that* matches *a gap-maximal substring v of $\mathcal{B}_s$. By "matching" we mean that $v[j] \subseteq u_i[j]$ for all $0 \leq j < |v|$.*

We are now ready to state our main theorem.

Theorem 3. *Let $s \in \Sigma^+$, let w be a motif of s provided online, and let $w = u_0 \bullet^{d_0} u_1 \bullet^{d_1} \cdots \bullet^{d_{k-2}} u_{k-1}$ be the gap-factorization of w induced by s. After a $O(|s|^3)$ offline preprocessing of s, we can compute $\sharp_{w,a}[i]$ for all $i \in \{1, \dots, |w|-1\}$ and all $a \in \Sigma$ in worst-case $O\left(\sum_{i=0}^{k-1} \sum_{j=i+1}^{k-1} (|u_i| + |u_j|) \log(|u_i| + |u_j|) + k|w|\right)$ time.*

Proof. Build $\mathcal{B}_s = \{t_0, t_1, \dots, t_{|\mathcal{B}_s|-1}\}$ in $O(|s|^2 \log |s|)$ time [126]: the result is a set of $O(|s|)$ tiling motifs of length $O(|s|)$ each [126]. For all $i \in \{0, \dots, |\mathcal{B}_s| - 1\}$, compute the convolution of t_i with itself in $O(|s|^2 \log |s|)$ time overall. Build matrices $\mathcal{P}_{t_i,a}$ and $\mathcal{S}_{t_i,a}$ for every $a \in \Sigma$ using Lemma 7, in overall $\sum_{i=0}^{|\mathcal{B}_s|} ||t_i|| \cdot |t_i| \in O(|s|^3)$ time and space. At the same cost, build the matrices used by Lemma 10. Then, build in $O(|s|^2)$ time the generalized suffix tree $\mathcal{T}_s$ of the strings in $\mathcal{B}_s$, treating $\bullet$ as different from every other symbol in Σ. In what follows, we will decorate the nodes of $\mathcal{T}_s$ with additional information that will help answering online queries. For clarity, given a tree $\mathcal{T}$, we will denote with $\mathcal{T}[\alpha]$ the value stored at node α of $\mathcal{T}$. First, we initialize a digital search tree $\mathcal{Q}_s$ with height two. For every tiling motif $t_i \in \mathcal{B}_s$, let $\mathcal{T}_i$ be its corresponding suffix tree. We set $\mathcal{T}_i[\alpha] = j$ for every node α in $\mathcal{T}_i$, where j is a position at which the substring of t_i associated with α occurs in t_i. This can be done in $O(|s|)$ time. Then, we mark all nodes α of $\mathcal{T}_s$ that correspond to nodes of $\mathcal{T}_i$ in $O(|s^2|)$ time, by traversing $\mathcal{T}_i$ and $\mathcal{T}_s$ top-down in parallel. Let α be a node of $\mathcal{T}_s$

that corresponds to node $\bar{\alpha}$ in $\mathcal{T}_i$: we set $\mathcal{T}_s[\alpha] = (i, \mathcal{T}_i[\bar{\alpha}])$. Similarly, let α and β be two nodes of $\mathcal{T}_s$ that correspond to nodes $\bar{\alpha}$ and $\bar{\beta}$ in $\mathcal{T}_i$, respectively. Then, we add to $\mathcal{Q}_s$ strings $\alpha\beta$ and $\beta\alpha$, and we store at the corresponding leaves of $\mathcal{Q}_s$ the triplet $(i, \mathcal{T}_i[\bar{\alpha}], \mathcal{T}_i[\bar{\beta}])$. This can be done in $O(|s|^2)$ time. $\mathcal{T}_i$ is then discarded and we proceed to the next i. The overall preprocessing of s thus takes $O(|s|^3)$ time and space.

Let now w be a motif of s provided online. Follow w in $\mathcal{T}_s$: if w is a substring of $\mathcal{B}_s$, then it has a proper locus α in $\mathcal{T}_s$, and $\mathcal{T}_s[\alpha]$ is sufficient to compute $\sharp_{w,a}[i]$ for all i and a in $O(|w|)$ time using Lemma 8. If w is not a substring of $\mathcal{B}_s$, then there is a position $0 \leq i < |w|$ such that $w[i] = \bullet$ and $\bullet$ cannot be found at the current position in the suffix tree. If the current position in the suffix tree lies inside an edge, we can continue matching w until such a mismatch happens at a node of the tree, i.e. until we find a node without symbol $\bullet$ among its children: this node corresponds to string v_0, the longest dot-maximal substring of $\mathcal{B}_s$ that matches prefix u_0 of w. We then continue reading from the next solid character of w starting from the root of $\mathcal{T}_s$, thus finding substrings $v_1, v_2, \ldots, v_k$ of $\mathcal{B}_s$ that match $u_1, u_2, \ldots, u_k$, respectively. In the worst case, the value of $\sharp_{u_i,a}[j]$ for $1 \leq j < |u_i|$ can be computed in $O(|u_i| \log |u_i|)$ time from the information stored at the node of $\mathcal{T}_s$ that corresponds to v_i, using Lemma 9. To compute $\sharp_{w,a}[i]$ we also need to know $\sharp_{u_i,u_j,a}[h]$ for $0 \leq i < j < k$. Let α_i and α_j be the nodes of $\mathcal{T}_s$ that correspond to v_i and v_j, respectively. Fact 2 and the structure of $\mathcal{T}_s$ guarantee that there is at least one string in $\mathcal{B}_s$ in which both v_i and v_j occur, thus string $\alpha_i\alpha_j$ must occur in $\mathcal{Q}_s$: using the information returned by $\mathcal{Q}_s$ and Lemma 8, we can thus access $\sharp_{v_i,v_j,a}[h]$ in constant time for any h. The value of $\sharp_{u_i,u_j,a}[h]$ can then be derived using natural adaptations of Lemmas 9 and 10 to pairs of strings. □

This construction fails for patterns that are *not* motifs of s, because such patterns are not capable of producing valid indexes in $\mathcal{Q}_s$. For example, assume that w is not a motif of s because, at some position i, it has a solid character that cannot be found at the corresponding position in $\mathcal{T}_s$. Assume that we terminate the current

factor at this point, and that we restart reading w from the root of $\mathcal{T}_s$, ending up with a factorization $u_0u_1 \dots u_{k-1}$, $u_i \in (\Sigma \cup \{\bullet\})^+$ for all i, not dissimilar to the one described above. Since w is not the sparsification of a substring of a tiling motif of s, we are not guaranteed that there is a tiling motif in which both u_i and u_j occur for all $0 \leq i < j < k$: accessing $\mathcal{Q}_s$ with the identifiers of the corresponding nodes in $\mathcal{T}_s$ could thus return no result. The same happens for patterns that are too long to be motif of s. Adapting our data structures to address this issue would likely require $\Omega(|s|^4)$ preprocessing, which tends to be impractical in most applications.

For arbitrary motifs, the worst-case performance of Theorem 3 is never asymptotically faster than computing the convolution of motif w with itself: for example, if k is bounded by a constant, the worst-case running time is $O(|w| \log |w|)$; if k is $O(\log |w|)$ the worst-case running time is $O(|w|(\log |w|)^3)$; and if k is $O(|w|)$, the worst-case running time is $O(|w|^2)$. However, Theorem 3 does link the time to process a motif w to the structure of w and of the text s. The following corollary, that derives immediately from Theorem 3 and Lemma 10, shows that motifs with a specific structure are amenable to particularly fast processing.

Corollary 4. *Let s be a string, let w be a motif of s, let $w = u_0 \bullet^{d_0} u_1 \bullet^{d_1} \cdots \bullet^{d_{k-2}} u_{k-1}$ be the gap-factorization of w induced by s, and let $v_0, v_1, \dots, v_{k-1}$ be the corresponding substrings of $\mathcal{B}_s$ that match the factors of w. A* block *in a factor u_i, $0 \leq i < k$, is a substring $u_i[d, d+\ell-1]$ such that $(u_i[j] = \bullet) \wedge (v_i[j] \neq \bullet)$ for $j = d$ and $j = d+\ell-1$, and such that $u_i[j] = \bullet$ for $d < j < d+\ell-1$. A block is* maximal *if it is not contained in any other block. If b_i, the number of maximal blocks in factor u_i, is bounded by a constant for all i, then we can compute $\sharp_{w,a}[i]$ for all $i \in \{1, \dots, |w|-1\}$ and all $a \in \Sigma$ in $O(k|w|)$ time after $O(|s|^3)$ preprocessing. Similarly, if k is bounded by a constant, then we can compute $\sharp_{w,a}[i]$ for all $i \in \{1, \dots, |w|-1\}$ and all $a \in \Sigma$ in $O(|w| \sum_{i=0}^{k-1} b_i)$ time after $O(|s|^3)$ preprocessing.*

Assuming that the number of maximal blocks in w is bounded by a constant is

probably not too restrictive in practice, since input motifs are likely to have a number of *maximal runs of gaps* bounded by an application-dependent constant.

3.5 Conclusion

Speeding up the computation of statistical properties of gapped patterns is crucial in large-scale molecular biology and text mining. The implicit technique used in Section 3.3 to bring the computation of the expectation and variance of a gapped pattern w from $O(2^{|w|})$ to $O(|w|^2)$ time has the desirable practical effect of limiting string operations to the construction of selectors, thus keeping only matrix and vector operations in the kernels. This likely allows to take better advantage of existing software libraries and hardware in a practical implementation of such equations. Lemma 6 is likely to be applicable to other statistical measures as well, for example to the *conditional* expectation and variance of a pattern given the occurrences of others (see e.g. [26]), and to the expectation and variance of a *set* of patterns allowed to overlap each other. Even the construction in Theorem 3 lends itself to multiple levels of optimization and tuning, both in the offline and in the online part. Implementing such algorithms efficiently could thus be a first step towards the construction of a comprehensive, high-performance library for computing statistics on gapped patterns, a much needed tool in all fields that deal with large unstructured texts.

The setup in Theorem 3 heavily relies on having string s available offline. Assuming that even s is given online would be more realistic in applications like security and logging, and would probably require a completely different set of data structures and algorithms. The gap-factorization of a motif seems also a notion of independent interest, and resonates with ideas in conditional algorithmic information and data compression (e.g. [90]): it would be interesting to relate the time to compute the variance of a motif to a measure of mutual algorithmic information between the motif and the basis. Another natural extension of this work would be optimizing

the computation of the variance for *maximal motifs*: maximal motifs have no special status in our current construction, mainly because they are not singled out by Fact 2. However, maximal motifs in s could bear a relation to right-maximal substrings, or to other saturated constructs, in $\mathcal{B}_s$. Finally, embedding the efficient computation of expectation and variance into existing algorithms that generate *all* motifs of a string s from its tiling basis $\mathcal{B}_s$ (e.g. [123]) could also be a stimulating extension.

CHAPTER IV

CONCLUSION AND EXTENSIONS

This thesis explored a notion of information in biological sequences that, rather than relating to negentropy or compressibility, is based on the dictionaries of all combinatorial substructures of a specific kind, and on suitable extremal subsets of these dictionaries. This notion, rooted in subword complexity, comes particularly natural with the current surge of alignment-free algorithms for genome and proteome comparison. We have focused on substructures whose length and sparsity have never been explored before, i.e. subsequences and rigid gapped motifs with bounded minimum density and unbounded length. Our measures highlight previously unseen laws that relate subsequence composition to string length and to the maximum distance between consecutive symbols, across a range of structurally and functionally diverse polypeptides. Similar counts on extremely dense and extremely sparse gapped motifs are shown to achieve state-of-the-art phylogeny reconstruction of mitochondrial proteomes, implying that the composition of such structures encodes shared evolutionary information.

Strings with gaps are likely to carry more phylogenetic information than short, solid blocks in sequences subjected to a high rate of random mutation, and requiring long-range interactions for stability or function: viral genomes would thus be ideal candidates for testing whether motif-based distances can outperform string-based distances. The very fact that long and extremely sparse motifs carry phylogenetic signal resonates with the medium-range pair correlations that have been measured in proteomes up to length 800 [23], and with the long-range correlations up to hundred

of thousands amino acids apart, caused by gene duplication and genomic rearrangements [173]. Medium-range correlations themselves are likely to be a trace of gene duplication, while also capturing the three-dimensional structure of proteins. Such correlations could be put at the core of a new alignment-free methodology, and their fast computation at the genome scale would likely lend itself to the formulation of interesting algorithms.

Another natural extension of our experiments would be to replace string-theoretic gapped motifs with *biologically significant gapped patterns*. As mentioned in Section 2.1, the gapped patterns in PROSITE have been found to be selectively over- and under-represented in proteomes and in the translated intergenic regions of some genomes, but these preferences have not been applied to phylogeny construction yet [116, 184]. Intergenic occurrences of PROSITE patterns have been conjectured to be relics of ancient proteins that have been deactivated by mutation [184]: if this is indeed the case, taking into account such occurrences in phylogeny reconstruction could move related taxa closer to their common ancestor, improving classification. The idea of using biologically significant components in composition vectors is not new to phylogeny: genomes have been represented as vectors indexed by COG clusters, and containing the frequency (or just the presence or absence) of genes belonging to each cluster (see, e.g., [99, 100, 111]). A similar approach has been applied to SCOP folds and domains (see e.g. [29, 46, 63, 98, 99] for a small sampler), other natural "words" in the protein dictionary of an organism. However, none of these studies has considered gapped structures, and none of them has been validated on more than few dozen proteomes.

Another key observation that emerges from our experiments is that phylogenetic information is not equally distributed in the dictionary of motifs. Different parts of a dictionary could even carry different signals: for example, preliminary experiments with the two-stranded genomes of nine nodaviruses show that elementary motifs with

six or more solid characters group together strands that correspond to *homologous function*, while both existing string-based algorithms and elementary motifs with four solid characters or less separate such strands according to their phylogenetic origin (Figure 42). This seems to resonate with the well-known multiplicity of codes that are superimposed in biological sequences [129]. The experiments in Section 2.2.1 also show that we could limit ourselves to specific *subsets* of the dictionary of motifs in phylogeny reconstruction. Using a limited set of compositional features has both computational benefits in storage space and running time, and nontrivial implications in molecular evolution. In case of sparse motifs, using a reduced set of features is even imperative to scale to genomes. In this thesis, we considered systematic ways to reduce the size of pattern dictionaries, like restricting density and number of solid characters, isolating sets of compact generators with limited redundancy, and resorting to the dictionary of the LZWA family of online compressors, that allows a controlled proportion of gaps to be interspersed with solid characters. Measures on the gapped and ungapped dictionaries of LZWA from test proteomes are seen to reconstruct phylogenies of comparable quality to state of the art alignment-free algorithms. By validating these experiments on a large scale, we expect LZWA to keep achieving comparable quality to current k-mer approaches, while using significantly fewer features and even disregarding large portions of the input genomes.

Even in case classification quality with LZWA proves consistently inferior to other methods, this algorithm has the potential for a very fast implementation, and it could thus be used to build a fast initial approximation to the correct phylogeny. Moreover, the very structure of LZW prompts the development of further algorithmic extensions that enable the extraction of more complex or biologically meaningful patterns at approximately the same speed. Specifically, we could generalize LZW into a family of *pattern extraction algorithms* with the following properties: first, they must read the input string $s \in \Sigma^+$ online, maintaining a constant number of dictionaries that

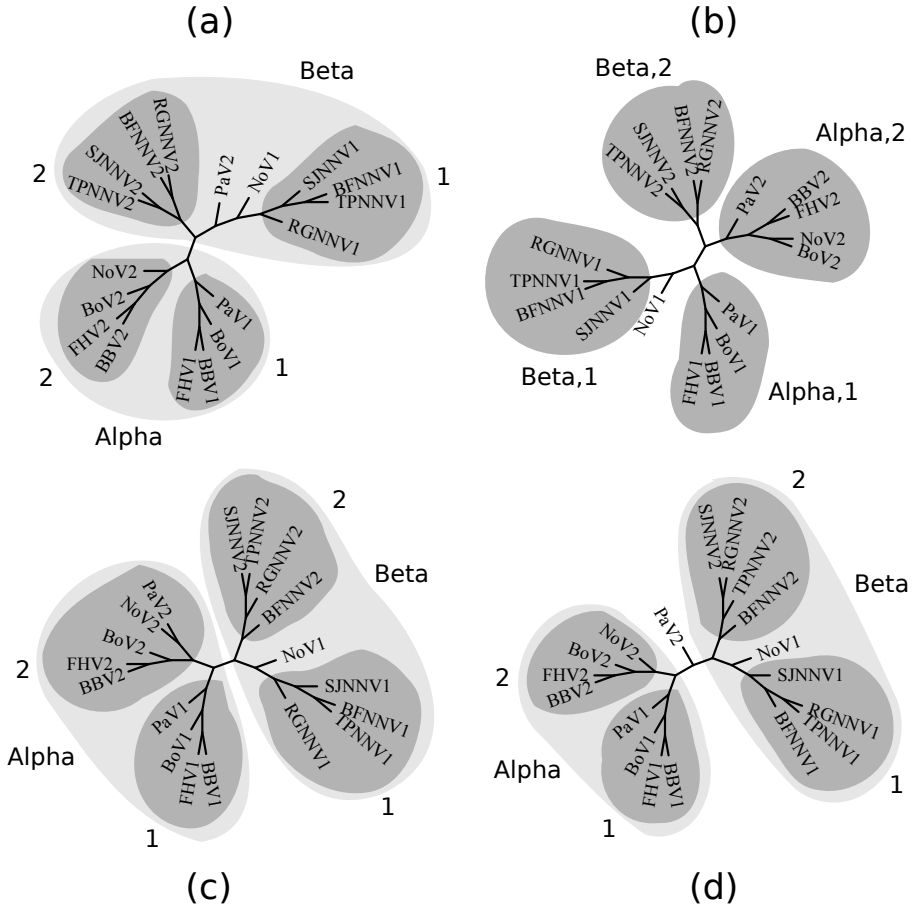

Figure 42: The composition of elementary motifs in nine nodaviruses. The genome of each virus consists of two RNA molecules: molecule one contains replication components, molecule two encodes the coat protein. Alpha nodaviruses primarily infect insects, Beta nodaviruses infect fish [80]. ACS (panel *c*) and CVTREE (panel *d*, $k = 5$) separate Alpha from Beta, then divide RNA1 from RNA2. Elementary motifs with 4 solid characters or less support the same subdivision (panel *a*: 4 solid characters, length between 48 and 50). However, elementary motifs with 5 solid characters generate random trees, and elementary motifs with 6, 7 and 8 solid characters separate RNA1 from RNA2 *and then* Alpha from Beta, detecting a functional similarity that is invisible with other methods (panel *b*: 6 solid characters, length between 45 and 50). Data from NCBI.

are updated with a constant number of elementary operations per input character. Second, they must be able to output a sequence σ of pointers to these dictionaries, with the requirement that there is an algorithm D which returns s on input σ. Since we are interested in pattern extraction rather than in compression, we do not require $|\sigma| \leq |s|$ in practice, nor in a significant fraction of Σ^+. Also, we do not require D to perform in linear time, nor σ to be stored somewhere in practice: the compressed string and the decoder are only there to ensure that all the information from s is stored in the dictionaries of the encoder, and that this information can be reversibly transformed to obtain s. A notable subset of this family of algorithms could, like LZW, have a decoder that proceeds online, and that synchronizes with the encoder after a constant number of steps. This setup brings to its logical extreme the notion of using patterns extracted by a compressor to measure similarity, which is at the core of algorithmic information.

The contributions of this thesis can be summarized as the following list of publications:

1. CUNIAL, F., "Faster variance computation for patterns with gaps," *Submitted*.
2. CUNIAL, F. and APOSTOLICO, A., "Phylogeny construction with rigid gapped motifs," *Journal of Computational Biology*. Accepted.
3. APOSTOLICO, A. and CUNIAL, F., "Sequence similarity by gapped LZW," in *Proceedings of the Data Compression Conference 2011*, pp. 343-352, March 2011.
4. APOSTOLICO, A. and CUNIAL, F., "The subsequence composition of polypeptides," *Journal of Computational Biology*, vol. 17, no. 8, pp. 1011-1049, 2010.
5. APOSTOLICO, A. and CUNIAL, F., "Probing the randomness of proteins by their subsequence composition," in *Proceedings of the Data Compression Conference 2009*, pp. 173-182, March 2009.
6. APOSTOLICO, A. and CUNIAL, F., "The subsequence composition of a string". *Theoretical Computer Science*, vol. 410, no. 43, pp. 4360-4371, 2009.

REFERENCES

[1] Adami, C. and Cerf, N., "Physical complexity of symbolic sequences," *Physica D: Nonlinear Phenomena*, vol. 137, pp. 62–69, 2000.

[2] Adjeroh, D. and Nan, F., "On compressibility of protein sequences," in *Proceedings of the Data Compression Conference 2006*, pp. 422–434, 2006.

[3] Amelio, A., Apostolico, A., and Rombo, S., "Image compression by 2D motif basis," in *Proceedings of the Data Compression Conference 2011*, pp. 153–162, March 2011.

[4] Anfinsen, C., "The formation and stabilization of protein structure," *Journal of Biochemistry*, vol. 128, pp. 737–749, 1972.

[5] Apostolico, A., "Of lempel-ziv-welch parses with refillable gaps," in *Proceedings of the Data Compression Conference 2005*, DCC '05, (Washington, DC, USA), pp. 338–347, IEEE Computer Society, 2005.

[6] Apostolico, A., "Fast gapped variants for Lempel–Ziv–Welch compression," *Information and Computation*, vol. 205, no. 7, pp. 1012–1026, 2007.

[7] Apostolico, A., "Maximal words in sequence comparisons based on subword composition," in *Algorithms and Applications*, vol. 6060 of *Lecture Notes in Computer Science*, pp. 34–44, Springer Berlin / Heidelberg, 2010.

[8] Apostolico, A., Bock, M., and Xu, X., "Annotated statistical indices for sequence analysis," in *Proceedings of the Compression and Complexity of Sequences 1997*, Sequences '97, (Washington, DC, USA), pp. 215–229, IEEE Computer Society, 1997.

[9] Apostolico, A., Bock, M., and Lonardi, S., "Monotony of surprise and large-scale quest for unusual words," in *Proceedings of the sixth annual international conference on Computational biology*, RECOMB '02, (New York, NY, USA), pp. 22–31, ACM, 2002.

[10] Apostolico, A., Bock, M., Lonardi, S., and Xu, X., "Efficient detection of unusual words," *Journal of Computational Biology*, vol. 7, no. 1, pp. 71–94, 2000.

[11] Apostolico, A., Comin, M., and Parida, L., "Conservative extraction of over-represented extensible motifs," *Bioinformatics*, vol. 21, pp. i9–i18, 2005.

[12] Apostolico, A., Comin, M., and Parida, L., "Mining, compressing and classifying with extensible motifs," *Algorithms for Molecular Biology*, vol. 1, 2006.

[13] Apostolico, A., Comin, M., and Parida, L., "VARUN: Discovering extensible motifs under saturation constraints," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 7, no. 4, pp. 752–726, 2010.

[14] Apostolico, A., Denas, O., and Dress, A., "Efficient tools for comparative substring analysis," *Journal of Biotechnology*, vol. 149, no. 3, pp. 120–126, 2010.

[15] Apostolico, A. and Ehrenfeucht, A., "Efficient detection of quasiperiodicities in strings," *Theoretical Computer Science*, vol. 119, no. 2, pp. 247–265, 1993.

[16] Apostolico, A. and Parida, L., "Compression and the wheel of fortune," in *Proceedings of the Data Compression Conference 2003*, pp. 143–152, March 2003.

[17] Apostolico, A. and Parida, L., "Incremental paradigms of motif discovery," *Journal of Computational Biology*, vol. 11, pp. 15–25, 2004.

[18] Apostolico, A. and Pizzi, C., "Monotone scoring of patterns with mismatches," in *In Proceedings of WABI 2004*, pp. 87–98, Springer, 2004.

[19] Apostolico, A. and Tagliacollo, C., "Optimal offline extraction of irredundant motif bases," in *Computing and Combinatorics* (Lin, G., ed.), vol. 4598 of *Lecture Notes in Computer Science*, pp. 360–371, Springer Berlin/Heidelberg, 2007.

[20] Atallah, M., Gwadera, R., and Szpankowski, W., "Detection of significant sets of episodes in event sequences," in *Proceedings of the Fourth IEEE International Conference on Data Mining*, ICDM '04, (Washington, DC, USA), pp. 3–10, IEEE Computer Society, 2004.

[21] Atteson, K., "Calculating the exact probability of language-like patterns in biomolecular sequences," in *ISMB-98 Proceedings*, pp. 17–24, 1998.

[22] Ben-Hur, A. and Brutlag, D., "Remote homology detection: a motif based approach," *Bioinformatics*, vol. 19, no. suppl. 1, pp. i26–i33, 2003.

[23] Benedetto, D., Caglioti, E., and Chica, C., "Compressing proteomes: the relevance of medium range correlations," *EURASIP Journal of Bioinformatics and Systems Biology*, p. 60723, 2007.

[24] Benson, G. and Waterman, M., "A method for fast database search for all k-nucleotide repeats," *Nucleic Acids Research*, vol. 22, pp. 4828–4836, 1994.

[25] Blaidsell, B., "A measure of the similarity of sets of sequences not requiring sequence alignment," in *Proceedings of the National Academy of Sciences*, vol. 83, pp. 5155–5159, 1986.

[26] Blanchette, M. and Sinha, S., "Separating real motifs from their artifacts," *Bioinformatics*, vol. 17, pp. S30–S38, 2001.

[27] Brooks, Jr., F. P., "Three great challenges for half-century-old computer science," *Journal of the ACM*, vol. 50, no. 1, pp. 25–26, 2003.

[28] Broome, B. and Hecht, M., "Nature disfavors sequences of alternating polar and nonpolar amino acids," *Journal of Molecular Biology*, vol. 296, pp. 961–968, 2000.

[29] Caetano-Anollés, G. and Caetano-Anollés, D., "Universal sharing patterns in proteomes and evolution of protein fold architecture and life," *Journal of Molecular Evolution*, vol. 60, pp. 484–498, 2005.

[30] Califano, A., "SPLASH: structural pattern localization analysis by sequential histograms," *Bioinformatics*, vol. 16, pp. 341–357, 2000.

[31] Cao, M., Dix, T., Allison, L., and Mears, C., "A simple statistical algorithm for biological sequence compression," in *Proceedings of the Data Compression Conference 2007*, pp. 43–52, 2007.

[32] Carothers, J. M., Oestreich, S. C., Davis, J. H., and Szostak, J. W., "Informational complexity and functional activity of RNA structures," *Journal of the American Chemical Society*, vol. 126, pp. 5130–5137, 2004.

[33] Chase, P. J., "Subsequence numbers and logarithmic concavity," *Discrete Mathematics*, vol. 16, no. 2, pp. 123–140, 1976.

[34] Chattaraj, A. and Parida, L., "VARUN: an inexact-suffix tree based algorithm for detecting extensible patterns," *Theoretical Computer Science*, vol. 335, 2005.

[35] Chor, B., Horn, D., Goldman, N., Levy, Y., and Massingham, T., "Genomic DNA k-mer spectra: models and modalities," *Genome Biology*, vol. 10, no. 10, p. R108, 2009.

[36] Chu, K. H., Qi, J., Yu, Z.-G., and Anh, V., "Origin and phylogeny of chloroplasts revealed by a simple correlation analysis of complete genomes," *Molecular Biology and Evolution*, vol. 21, no. 1, pp. 200–206, 2004.

[37] Cilibrasi, R. and Vitányi, P., "Clustering by compression," *IEEE Transactions on Information Theory*, vol. 51, pp. 1523–1545, 2005.

[38] Cole, R. and Hariharan, R., "Verifying candidate matches in sparse and wildcard matching," in *Proceedings of the thiry-fourth annual ACM Symposium on Theory of Computing*, STOC '02, (New York, NY, USA), pp. 592–601, ACM, 2002.

[39] Colosimo, A. and De Luca, A., "Special factors in biological strings," *Journal of Theoretical Biology*, vol. 58, pp. 29–46, 2000.

[40] Comin, M. and Parida, L., "Detection of subtle variations as consensus motifs," *Theoretical Computer Science*, vol. 395, pp. 158–170, 2008.

[41] Comin, M. and Verzotto, D., "Classification of protein sequences by means of irredundant patterns," *BMC Bioinformatics*, vol. 11, no. S16, 2010.

[42] Comin, M. and Verzotto, D., "The irredundant class method for remote homology detection of protein sequences," *Journal of Computational Biology*, vol. 18, no. 12, pp. 1–11, 2011.

[43] Darzentas, N. and Rigoutsos, I., "Sensitive detection of sequence similarity using combinatorial pattern discovery: a challenging study of two distantly related protein families," *Proteins*, vol. 61, pp. 926–937, 2005.

[44] Davidson, A., Lumb, K., and Sauer, R., "Cooperatively folded proteins in random sequence libraries," *Nature Structural Biology*, vol. 2, pp. 856–864, 1995.

[45] Davidson, A. and Sauer, R., "Folded proteins occur frequently in libraries of random amino acid sequences," *PNAS*, vol. 91, pp. 2146–2150, 1994.

[46] Deeds, E., Hennessey, H., and Shakhnovich, E., "Prokaryotic phylogenies inferred from protein structural domains," *Genome Research*, vol. 15, pp. 393–402, 2005.

[47] Doi, N., Kakukawa, K., Oishi, Y., and Yanagawa, H., "High solubility of random-sequence proteins consisting of five kinds of primitive amino acids," *Protein Engineering*, vol. 18, pp. 279–284, 2005.

[48] Dong, Q., Wang, X., and Lin, L., "Application of latent semantic analysis to protein remote homology detection," *Bioinformatics*, vol. 22, no. 3, pp. 285–290, 2006.

[49] Dufton, M., "Genetic code synonym quotas and amino acid complexity: cutting the cost of proteins?," *Journal of Theoretical Biology*, vol. 187, pp. 165–173, 1997.

[50] Edgar, R., "Local homology recognition and distance measures in linear time using compressed amino-acid alphabets," *Bioinformatics*, vol. 32, pp. 380–385, 2004.

[51] Edwards, R., Davey, N., and Shields, D., "SLiMFinder: a probabilistic method for identifying over-represented, convergently evolved, short linear motifs in proteins," *PLoS ONE*, vol. 2, p. e967, 10 2007.

[52] Elzinga, C., Rahmann, S., and Wang, H., "Algorithms for subsequence combinatorics," *Theoretical Computer Science*, vol. 409, no. 3, pp. 394–404, 2008.

[53] Elzinga, C., "Complexity of categorical time series," *Sociological Methods and Research*, vol. 38, no. 3, pp. 463–481, 2010.

[54] Felsenstein, J., "PHYLIP (Phylogeny Inference Package) version 3.6." Distributed by the author, 2005. Department of Genome Sciences, University of Washington, Seattle.

[55] Ferragina, P., Giancarlo, R., Greco, V., Manzini, G., and Valiente, G., "Compression-based classification of biological sequences and structures via the Universal Similarity Metric: experimental assessment," *BMC Bioinformatics*, vol. 8, pp. 252–272, 2007.

[56] Ferreira, P. and Azevedo, P., "Evaluating deterministic motif significance measures in protein databases," *Algorithms for Molecular Biology*, vol. 2, no. 1, p. 16, 2007.

[57] Fischer, M. and Paterson, M., "String-matching and other products," tech. rep., Massachusetts Institute of Technology, Cambridge, MA, USA, 1974.

[58] Flajolet, P., Guivarc'h, Y., Szpankowski, W., and Vallée, B., "Hidden pattern statistics," in *Proceedings of the 28th International Colloquium on Automata, Languages and Programming,*, ICALP '01, (London, UK), pp. 152–165, Springer-Verlag, 2001.

[59] Flaxman, A., Sorkin, G. B., and Harrow, A. W., "Strings with maximally many distinct subsequences and substrings," *Electronic Journal of Combinatorics*, vol. 8, 2004.

[60] Fredman, M. and Tarjan, R., "Fibonacci heaps and their uses in improved network optimization algorithms," *Journal of the ACM*, vol. 34, pp. 596–615, July 1987.

[61] Galas, D., Nykter, M., Carter, G., Price, N., and Shmulevich, I., "Biological information as set-based complexity," *IEEE Transactions on Information Theory*, vol. 56, pp. 667–677, February 2010.

[62] Gallé, M., *Searching for compact hierarchical structures in DNA by means of the Smallest Grammar Problem*. PhD thesis, Université de Rennes 1, 2011.

[63] Gerstein, M., "Patterns of protein-fold usage in eight microbial genomes: A comprehensive structural census," *Proteins: Structure, Function, and Bioinformatics*, vol. 33, no. 4, pp. 518–534, 1998.

[64] Giancarlo, R., Scaturro, D., and Utro, F., "Textual data compression in computational biology: a synopsis," *Bioinformatics*, vol. 25, pp. 1575–1586, 2009.

[65] Gould, C., "ELM: the status of the 2010 Eukaryotic Linear Motif resource," *Nucleic Acids Research*, pp. 1–14, 2009.

[66] Gratzer, G., *General Lattice Theory*. Birkhauser, 2nd ed., 1998.

[67] Grossi, R., Pietracaprina, A., Pisanti, N., Pucci, G., Upfal, E., and Vandin, F., "MADMX – a strategy for maximal dense motif extraction," *Journal of Computational Biology*, vol. 18, no. 4, pp. 535–545, 2011.

[68] Gwadera, R., Atallah, M., and Szpankowski, W., "Reliable detection of episodes in event sequences," in *Knowledge and Information Systems*, pp. 67–74, 2004.

[69] Gwadera, R., Atallah, M., and Szpankowski, W., *Markov models for identification of significant episodes*, pp. 404–414. SIAM, 2005.

[70] Gwadera, R. and Crestani, F., "Discovering significant patterns in multi-stream sequences," in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, (Washington, DC, USA), pp. 827–832, IEEE Computer Society, 2008.

[71] Han, K. and Baker, D., "Recurring local sequence motifs in proteins," *Journal of Molecular Biology*, vol. 251, pp. 176–187, 1995.

[72] Han, K., Bystroff, C., and Baker, D., "Three-dimensional structures and contexts associated with recurrent amino acid sequence patterns," *Protein Science*, vol. 6, pp. 1587–1590, 1997.

[73] Hao, B. and Qi, J., "Procaryote phylogeny without sequence alignment: from avoidance signature to composition distance," *Journal of Bioinformatics and Computational Biology*, vol. 2, pp. 1–19, 2004.

[74] Hart, R., Royyuru, A., Stolovitzky, G., and Califano, A., "Systematic and fully automated identification of protein sequence patterns," *Journal of Computational Biology*, vol. 7, no. 3/4, pp. 585–600, 2000.

[75] Hategan, A. and Tabus, I., "Protein is compressible," in *Proceedings of the 6th Nordic Signal Processing Symposium*, pp. 192–195, 2004.

[76] Hirschberg, D. and Regnier, M., "Tight bounds on the number of string subsequences," *Journal of Discrete Algorithms*, vol. 1, no. 1, pp. 123–132, 2000.

[77] Höhl, M., Rigoutsos, I., and Ragan, M., "Pattern-based phylogenetic distance estimation and tree reconstruction," *Evolutionary Bioinformatics Online*, vol. 2, pp. 359–375, 2006.

[78] Iliopoulos, C., Mohamed, M., Mouchard, L., Perdikuri, K., Smyth, W., and Tsakalidis, A., "String regularities with don't cares.," *Nordic Journal of Computing*, pp. 40–51, 2003.

[79] Jiménez-Montaño, M., "On the syntactic structure of protein sequences and the concept of grammar complexity," *Bulletin of Mathematical Biology*, vol. 46, pp. 641–659, 1984.

[80] Johnson, K., Johnson, K., Dasgupta, R., Gratsch, T., and Ball, L., "Comparisons among the larger genome segments of six nodaviruses and their encoded RNA replicases," *Journal of General Virology*, vol. 82, pp. 1855–1866, 2001.

[81] Jonassen, I., Collins, J., and Higgins, D., "Finding flexible patterns in unaligned protein sequences," *Protein Science*, vol. 4, pp. 1587–1595, 1995.

[82] Karlin, S., "Quantile distributions of amino acid usage in protein classes," *Protein Engineering*, vol. 5, pp. 729–738, 1992.

[83] Karlin, S. and Brendel, V., "Chance and statistical significance in protein and DNA sequence analysis," *Science*, vol. 257, no. 5066, pp. 39–49, 1992.

[84] Kása, Z., "On the d-complexity of strings," *Pure Mathematics and Applications*, vol. 9, no. 1–2, pp. 119–128, 1998.

[85] Kawashima, S., Pokarowski, P., Pokarowska, M., Kolinski, A., Katayama, T., and Kanehisa, M., "AAindex: amino acid index database, progress report 2008," *Nucleic Acids Research*, vol. 36, pp. D202–D205, 2008.

[86] Kleffe, J. and Borodovsky, M., "First and second moment of counts of words in random texts generated by Markov chains," *Bioinformatics/Computer Applications in the Biosciences*, vol. 8, pp. 433–441, 1992.

[87] Kolmogorov, A., "Three approaches to the quantitative definition of information," *International Journal of Computer Mathematics*, vol. 2, no. 1–4, pp. 157–168, 1968.

[88] Konorski, J. and Szpankowski, W., "What is information?," in *IEEE Information Theory Workshop, ITW '08.*, pp. 269–270, May 2008.

[89] Lapinsh, M., Gutcaits, A., Prusis, P., Post, C., Lundstedt, T., and Wikberg, J., "Classification of G-protein coupled receptors by alignment-independent extraction of principal chemical properties of primary amino acid sequences," *Protein Science*, vol. 11, pp. 795–805, 2002.

[90] Lempel, A. and Ziv, J., "On the complexity of finite sequences," *IEEE Transactions on Information Theory*, vol. 22, pp. 75–81, January 1976.

[91] Leslie, C. and Kuang, R., "Fast kernels for inexact string matching," in *Sixteenth Annual Conference on Learning Theory and Seventh Kernel Workshop*, pp. 114–128, 2003.

[92] Li, M., Badger, J., Chen, X., Kwong, S., Kearney, P., and Zhang, H., "An information-based sequence distance and its application to whole mitochondrial genome phylogeny," *Bioinformatics*, vol. 17, no. 2, pp. 149–154, 2001.

[93] Li, M., Chen, X., Li, X., Ma, B., and Vitányi, P., "The similarity metric," in *Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms*, pp. 863–872, 2003.

[94] Li, T., Fan, K., Wang, J., and Wang, W., "Reduction of protein sequence complexity by residue grouping," *Protein Engineering*, vol. 16, pp. 323–330, 2003.

[95] Li, W., "The complexity of DNA," *Complexity*, vol. 3, no. 2, 1997.

[96] Li, X., Obradovic, Z., Brown, C., Garner, E., and Dunker, A., "Comparing predictors of disordered protein," *Workshop on Genome Informatics*, vol. 11, pp. 172–184, 2000.

[97] Liao, H., Yeh, W., Chiang, D., Jernigan, R., and Lustig, B., "Protein sequence entropy is closely related to packing density and hydrophobicity," *Protein Engineering*, vol. 18, pp. 59–64, 2005.

[98] Lin, J. and Gerstein, M., "Whole-genome trees based on the occurrence of folds and orthologs: implications for comparing genomes on different levels," *Genome Research*, vol. 10, pp. 808–818, 2000.

[99] Lin, J., Qian, J., Greenbaum, D., Bertone, P., Das, R., Echols, N., Senes, A., Stenger, B., and Gerstein, M., "GeneCensus: genome comparisons in terms of metabolic pathway activity and protein family sharing," *Nucleic Acids Research*, vol. 30, pp. 4574–4582, 2002.

[100] Ling, L., Wang, J., Cui, Y., Li, W., and Chen, R., "Proteome-wide analysis of protein function composition reveals the clustering and phylogenetic properties of organisms," *Molecular Phylogenetics and Evolution*, vol. 25, no. 1, pp. 101–111, 2002.

[101] Lingner, T. and Meinicke, P., "Remote homology detection based on oligomer distances," *Bioinformatics*, vol. 22, no. 18, pp. 2224–2231, 2006.

[102] Liu, A. and Califano, A., "Functional classification of proteins by pattern discovery and top-down clustering of primary sequences," *IBM Systems Journal*, vol. 40, no. 2, pp. 379–393, 2001.

[103] Liu, A., Zhang, X., Stolovitzky, G., Califano, A., and Firestein, S., "Motif-based construction of a functional map for mammalian olfactory receptors," *Genomics*, vol. 81, pp. 443–456, 2003.

[104] Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., and Watkins, C., "Text classification using string kernels," *Journal of Machine Learning Research*, vol. 2, pp. 419–444, March 2002.

[105] Lonardi, S., *Global detectors of unusual words: design, implementation, and applications to pattern discovery in biosequences*. PhD thesis, Purdue University, West Lafayette, IN, USA, 2001.

[106] Lupas, A., "On the evolution of protein folds: are similar motifs in different protein folds the result of convergence, insertion, or relics of an ancient peptide world?," *Journal of Structural Biology*, vol. 134, pp. 191–203, 2001.

[107] Macchiato, M., Cuomo, V., and Tramontano, A., "Determination of the autocorrelation orders of proteins," *European Journal of Biochemistry*, vol. 149, pp. 375–379, 1985.

[108] Matsumoto, T., Sadakane, K., and Imai, H., "Biological sequence compression algorithms," *Genome Informatics*, vol. 11, pp. 43–52, 2000.

[109] McHardy, A., Martín, H., Tsirigos, A., Hugenholtz, P., and Rigoutsos, I., "Accurate phylogenetic classification of variable-length DNA fragments," *Nature Methods*, vol. 4, no. 1, pp. 63–72, 2007.

[110] Monod, J., *Chance and necessity*. Collins London, 1972.

[111] Montague, M. and Hutchison, C., "Gene content phylogeny of herpesviruses," *Proceedings of the National Academy of Sciences*, vol. 97, no. 10, pp. 5334–5339, 2000.

[112] Murzin, A., Brenner, S., Hubbard, T., and Chothia, C., "SCOP: a structural classification of proteins database for the investigation of sequences and structures," *Journal of Molecular Biology*, vol. 247, pp. 536–540, 1995.

[113] Nakai, K., Kidera, A., and Kanehisa, M., "Cluster analysis of amino acid indices for prediction of protein structure and function," *Protein Engineering*, vol. 2, pp. 93–100, 1988.

[114] Nevill-Manning, C. and Witten, I., "Protein is incompressible," in *Proceedings of the Data Compression Conference 1999*, p. 257, 1999.

[115] Nicodème, P., "Regexpcount, a symbolic package for counting problems on regular expressions and words," *Fundamenta Informaticae*, vol. 56, pp. 71–88, October 2002.

[116] Nicodème, P., Doerks, T., and Vingron, M., "Proteome analysis based on motif statistics," *Bioinformatics*, vol. 18, pp. S161–S171, 2002.

[117] Nicodème, P., Salvy, B., and Flajolet, P., "Motif statistics," *Theoretical Computer Science*, vol. 287, pp. 593–617, 2002.

[118] Orengo, C., Michie, A., Jones, S., Jones, D., Swindells, M., and Thornton, J., "CATH: a hierarchic classification of protein domain structures," *Structure*, vol. 5, pp. 1093–1109, 1997.

[119] Otu, H. and Sayood, K., "A new sequence distance measure for phylogenetic tree reconstruction," *Bioinformatics*, vol. 19, pp. 2122–2130, 2003.

[120] Pande, V., Grosberg, A., and Tanaka, T., "Nonrandomness in protein sequences: evidence for a physically driven stage of evolution," *PNAS*, vol. 91, pp. 12972–12975, 1994.

[121] Parida, L., Floratos, A., and Rigoutsos, I., "An approximation algorithm for alignment of multiple sequences using motif discovery," *Journal of Combinatorial Optimization*, vol. 3, no. 2–4, pp. 247–275, 1999.

[122] Parida, L., Rigoutsos, I., Floratos, A., Platt, D., and Gao, Y., "Pattern discovery on character sets and real-valued data: linear bound on irredundant motifs and an efficient polynomial time algorithm," in *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, SODA 2000, (Philadelphia, PA, USA), pp. 297–308, Society for Industrial and Applied Mathematics, 2000.

[123] Parida, L., Rigoutsos, I., and Platt, D., "An output-sensitive flexible pattern discovery algorithm," in *Proceedings of the 12th Annual Symposium on Combinatorial Pattern Matching*, CPM '01, (London, UK), pp. 131–142, Springer-Verlag, 2001.

[124] Pisanti, N., Carvalho, A., Marsan, L., and Sagot, M.-F., "RISOTTO: Fast extraction of motifs with mismatches," in *LATIN 2006: Theoretical Informatics* (Correa, J., Hevia, A., and Kiwi, M., eds.), vol. 3887 of *Lecture Notes in Computer Science*, pp. 757–768, Springer Berlin / Heidelberg, 2006.

[125] Pisanti, N., Crochemore, M., Grossi, R., and Sagot, M.-F., "Bases of motifs for generating repeated patterns with don't cares," tech. rep., University of Pisa, 2003.

[126] Pisanti, N., Crochemore, M., Grossi, R., and Sagot, M.-F., "A basis of tiling motifs for generating repeated patterns and its complexity for higher

quorum," in *Proceedings of the 28th Mathematical Foundations of Computer Science Symposium*, pp. 622–631, 2003.

[127] Pisanti, N., Crochemore, M., Grossi, R., and Sagot, M., "Bases of motifs for generating repeated patterns with wildcards," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 2, no. 1, pp. 40–50, 2005.

[128] Pizzi, C. and Bianco, M., "Expectation of strings with mismatches under markov chain distribution," in *Proceedings of the 16th International Symposium on String Processing and Information Retrieval*, SPIRE '09, pp. 222–233, 2009.

[129] Popov, O., Segal, D., and Trifonov, E., "Linguistic complexity of protein sequences as compared to texts of human languages," *Biosystems*, vol. 38, no. 1, pp. 65–74, 1996.

[130] Przytycka, T., Srinivasan, R., and Rose, G., "Recursive domains in proteins," *Protein Science*, vol. 11, pp. 409–417, 2002.

[131] Ptitsyn, O. and Volkenstein, M., "Protein structure and neutral theory of evolution," *Journal of Biomolecular Structure and Dynamics*, vol. 4, pp. 137–56, 1986.

[132] Qi, J., Wang, B., and Hao, B., "Whole proteome prokaryote phylogeny without sequence alignment: a k-string composition approach," *Journal of Molecular Evolution*, vol. 58, pp. 1–11, 2004.

[133] Rackovsky, S., "Hidden sequence periodicities and protein architecture," *PNAS*, vol. 95, pp. 8580–8584, 1998.

[134] Rahman, R. and Rackovsky, S., "Protein sequence randomness and sequence-structure correlations," *Biophysical Journal*, vol. 68, pp. 1531–1539, 1995.

[135] Reinert, G., Schbath, S., and Waterman, M., "Probabilistic and statistical properties of words: an overview," *Journal of Computational Biology*, vol. 7, pp. 1–46, 2000.

[136] Richardson, J., "The anatomy and taxonomy of protein structure," *Advances in Protein Chemistry*, vol. 34, pp. 167–339, 1981.

[137] Rigoutsos, I. and Floratos, A., "Combinatorial pattern discovery in biological sequences: the TEIRESIAS algorithm," *Bioinformatics*, vol. 14, no. 1, pp. 55–67, 1998.

[138] Rigoutsos, I., Floratos, A., Ouzounis, C., Gao, Y., and Parida, L., "Dictionary building via unsupervised hierarchical motif discovery in the sequence space of natural proteins," *Proteins*, vol. 37, no. 2, pp. 264–277, 1999.

[139] Rigoutsos, I., Gao, Y., Floratos, A., and Parida, L., "Building dictionaries of 1D and 3D motifs by mining the unaligned 1D sequences of 17 archaeal and bacterial genomes," in *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, 1999.

[140] Rigoutsos, I., Huynh, T., Floratos, A., Parida, L., and Platt, D., "Dictionary-driven protein annotation," *Nucleic Acids Research*, vol. 30, no. 17, pp. 3901–3916, 2002.

[141] Robin, S. and Daudin, J.-J., "Exact distribution of word occurrences in a random sequence of letters," *Journal of Applied Probability*, vol. 36, no. 1, pp. 179–193, 1999.

[142] Robin, S., Daudin, J.-J., and Bernard, R., "Exact distribution of the distances between any occurrences of a set of words," *Annals of the Institute of Statistical Mathematics*, vol. 36, pp. 895–905, 2001.

[143] Robin, S., Daudin, J.-J., Richard, H., Sagot, M.-F., and Schbath, S., "Occurrence probability of structured motifs in random sequences," *Journal of Computational Biology*, pp. 761–774, 2002.

[144] Robinson, D. and Foulds, L., "Comparison of phylogenetic trees," *Mathematical Biosciences*, vol. 53, no. 1–2, pp. 131–147, 1981.

[145] Romero, P., Obradovic, Z., and Dunker, A., "Folding minimal sequences: the lower bound for sequence complexity of globular proteins," *FEBS Letters*, vol. 462, pp. 363–367, 1999.

[146] Romero, P., Obradovic, Z., Li, X., Garner, E., Brown, C., and Dunker, A., "Sequence complexity of disordered protein," *Proteins*, vol. 42, pp. 38–48, 2000.

[147] Rost, B., "Did evolution leap to create the protein universe?," *Current Opinion in Structural Biology*, vol. 12, pp. 409–416, 2002.

[148] Rousu, J. and Jaakkola, T., "Efficient computation of gapped substring kernels on large alphabets," *Journal of Machine Leaning Research*, vol. 6, pp. 1323–1344, 2005.

[149] S. Lifson and C. Sander, "Antiparallel and parallel β-strands differ in amino acid residue preferences," *Nature*, vol. 282, pp. 109–111, 1979.

[150] Sampath, G., "A block coding method that leads to significantly lower entropy values for the proteins and coding sections of *Haemophilus influenzae*," in *Proceedings of the IEEE Computer Society Conference on Bioinformatics*, p. 287, 2003.

[151] Schachtel, G., Bucher, P., Mocarski, E., Blaisdell, B., and Karlin, S., "Evidence for selective evolution in codon usage in conserved amino acid segments of human alphaherpesvirus proteins," *Journal of Molecular Evolution*, vol. 33, pp. 483–494, 1991.

[152] Schwartz, R. and King, J., "Frequencies of hydrophobic and hydrophilic runs and alternations in proteins of known structure," *Protein Science*, vol. 15, pp. 102–112, 2006.

[153] Sculley, D. and Brodley, C., "Compression and machine learning: a new perspective on feature space vectors," in *Proceedings of the Data Compression Conference 2006*, pp. 332–341, 2006.

[154] Sickmeier, M., Hamilton, J., LeGall, T., Vacic, V., Cortese, M., Tantos, A., Szabo, B., Tompa, P., Chen, J., Uversky, V., Obradovic, Z., and Dunker, A., "DisProt: the database of disordered proteins," *Nucleic Acids Research*, vol. 35, pp. D786–D793, 2007.

[155] Sigrist, C., Cerutti, L., de Castro, E., Langendijk-Genevaux, P., Bulliard, V., Bairoch, A., and Hulo, N., "PROSITE, a protein domain database for functional characterization and annotation," *Nucleic Acids Research*, vol. 38, pp. 161–6, 2010.

[156] Sims, G., Juna, S., Wu, G., and Kim, S., "Alignment-free genome comparison with feature frequency profiles (FFP) and optimal resolutions," *PNAS*, vol. 106, pp. 2677–2682, 2009.

[157] Sinha, S. and Tompa, M., "A statistical method for finding transcription factor binding sites," in *Proceedings of the International Conference on Intelligent Systems for Molecular Biology*, vol. 8, pp. 344–354, 2000.

[158] Sinha, S. and Tompa, M., "Discovery of novel transcription factor binding sites by statistical overrepresentation," *Nucleic Acids Research*, vol. 30, no. 24, pp. 5549–5560, 2002.

[159] Sinha, S. and Tompa, M., "YMF: a program for discovery of novel transcription factor binding sites by statistical overrepresentation," *Nucleic Acids Research*, vol. 31, no. 13, pp. 3586–3588, 2003.

[160] Sosinsky, A., Honig, B., Mann, R., and Califano, A., "Discovering transcriptional regulatory regions in *Drosophila* by a nonalignment method for phylogenetic footprinting," *PNAS*, vol. 104, no. 15, pp. 6305–6310, 2007.

[161] Stolovitzky, G. and Califano, A., "Statistical significance of patterns in biosequences," *IBM research report*, 1998.

[162] Strait, B. and Dewey, T., "The Shannon information entropy of protein sequences," *Biophysical Journal*, vol. 71, pp. 148–155, 1996.

[163] Stuart, G., Moffett, K., and Baker, S., "Integrated gene and species phylogenies from unaligned whole genome protein sequences," *Bioinformatics*, vol. 18, no. 1, pp. 100–108, 2002.

[164] Stuart, G., Moffett, K., and Leader, J., "A comprehensive vertebrate phylogeny using vector representations of protein sequences from whole genomes," *Molecular Biology and Evolution*, vol. 19, no. 4, pp. 554–562, 2002.

[165] Tsirigos, A. and Rigoutsos, I., "A new computational method for the detection of horizontal gene transfer events," *Nucleic Acids Research*, vol. 33, no. 3, pp. 922–933, 2005.

[166] Tsirigos, A. and Rigoutsos, I., "Human and mouse introns are linked to the same processes and functions through each genome's most frequent non-conserved motifs," *Nucleic Acids Research*, vol. 36, no. 10, pp. 3484–3493, 2008.

[167] Ulitsky, I., Burstein, D., Tuller, T., and Chor, B., "The average common substring approach to phylogenomic reconstruction," *Journal of Computational Biology*, vol. 13, pp. 336–350, 2006.

[168] van Helden, J., "Metrics for comparing regulatory sequences on the basis of pattern counts," *Bioinformatics*, vol. 20, pp. 399–406, 2004.

[169] Vinga, S., "Biological sequence analysis by vector-valued functions: revisiting alignment-free methodologies for DNA and protein classification," in *Advanced Computational Methods for Biocomputing and Bioimaging* (Pham, T., Yan, H., and Crane, D., eds.), New York: Nova Science Publishers, 2007.

[170] Vinga, S. and Almeida, J., "Alignment-free sequence comparison: a review," *Bioinformatics*, vol. 19, pp. 513–523, 2003.

[171] Wan, H. and Wootton, J., "A global compositional complexity measure for biological sequences," *Computers and Chemistry*, vol. 24, pp. 71–94, 2000.

[172] Weathers, E., Paulaitis, M., Woolf, T., and Hoh, J., "Insights into protein structure and function from disorder-complexity space," *Proteins*, vol. 66, pp. 16–28, 2006.

[173] Weiss, O. and Herzel, H., "Correlations in protein sequences and property codes," *Journal of Theoretical Biology*, vol. 190, pp. 341–353, 1998.

[174] Weiss, O., Jiménez-Montaño, M., and Herzel, H., "Information content of protein sequences," *Journal of Theoretical Biology*, vol. 206, pp. 379–386, 2000.

[175] Welch, T., "A technique for high-performance data compression," *IEEE Computer*, vol. 17, no. 6, pp. 8–19, 1984.

[176] White, S., "Global statistics of protein sequences: implications for the origin evolution and prediction of structure," *Annual Review of Biophysics and Biomolecular Structure*, vol. 23, pp. 407–439, 1994.

[177] White, S. and Jacobs, R., "Statistical distribution of hydrophobic residues along the length of protein chains," *Biophysical Journal*, vol. 57, pp. 911–921, 1990.

[178] Wise, M., "0j.py: a software tool for low complexity proteins and protein domains," *Bioinformatics*, vol. 17, pp. 288–295, 2001.

[179] Wootton, J., "Sequences with unusual amino acid compositions," *Current Opinion in Structural Biology*, vol. 4, pp. 413–421, 1994.

[180] Wright, P. and Dyson, H., "Intrinsically unstructured proteins: reassessing the protein structure-function paradigm," *Journal of Molecular Biology*, vol. 293, pp. 321–331, 1999.

[181] Wu, T., Bruke, J., and Davison, D., "A measure of DNA dissimilarity based on the Mahalanobis distance between frequencies of words," *Biometrics*, vol. 53, pp. 1431–1439, 1997.

[182] Xiao, X., Shao, S., Ding, Y., Huang, Z., Huang, Y., and Chou, K., "Using complexity measure factor to predict protein subcellular location," *Amino Acids*, vol. 28, pp. 57–61, 2005.

[183] Xu, Z. and Hao, B., "CVTree update: a newly designed phylogenetic study platform using composition vectors and whole genomes," *Nucleic Acids Research*, vol. 37, pp. W174–W178, 2009.

[184] Zhang, Z., Harrison, P., and Gerstein, M., "Digging deep for ancient relics: a survey of protein motifs in the intergenic sequences of four eukaryotic genomes," *Journal of Molecular Biology*, vol. 323, pp. 811–822, 2002.

[185] Ziv, J. and Lempel, A., "A universal algorithm for sequential data compression," *IEEE Transactions on Information Theory*, vol. IT-23, no. 3, pp. 337–343, 1977.

[186] Ziv, J. and Lempel, A., "Compression of individual sequences via variable-rate coding," *IEEE Transactions on Information Theory*, vol. 24, no. 5, pp. 530–536, 1978.