

On the Monotonicity of the String Correction Factor for Words with Mismatches

(extended abstract)

Alberto Apostolico*

Cinzia Pizzi**

Georgia Tech & Univ. of Padova Univ. of Padova & Univ. of Helsinki

Abstract. The string correction factor is the term by which the probability of a word w needs to be multiplied in order to account for character changes or “errors” occurring in at most k arbitrary positions in that word. The behavior of this factor, as a function of k and of the word length, has implications on the number of candidates that need to be considered and weighted when looking for subwords of a sequence that present unusually recurrent replicas within some bounded number of mismatches. Specifically, it is seen that over intervals of mono- or bi-tonicity for the correction factor, only some of the candidates need be considered. This mitigates the computation and leads to tables of over-represented words that are more compact to represent and inspect. In recent work, expectation and score monotonicity has been established for a number of cases of interest, under *i.i.d.* probabilistic assumptions. The present paper reviews the cases of bi-tonic behavior for the correction factor, concentrating on the instance in which the question is still open.

1 Introduction

In computational biology, a *motif* is often described as a segment of DNA or proteins sequence carrying some functional or structural information. In the context of gene regulation, for instance, motifs are short sequences of DNA that belong to the region of the gene upstream the coding region. They represent the locations to which specific proteins, called *transcription factors*, bind to DNA to start the process of coding that will transform a segment of DNA into a protein. A common approach to the discovery of these regulatory sites goes through the detection of statistically significant regions. This pursues the commonly accepted hypothesis that patterns that are unusually frequent might carry some important

* Dipartimento di Ingegneria dell' Informazione, Università di Padova, Padova, Italy and College of Computing, Georgia Institute of Technology, 801 Atlantic Drive, Atlanta, GA 30318, USA. axa@dei.unipd.it Work Supported in part by the Italian Ministry of University and Research under the National Projects FIRB RBNE01KNFP, and PRIN “Combinatorial and Algorithmic Methods for Pattern Discovery in Biosequences”, and by the Research Program of the University of Padova.

** Dipartimento di Ingegneria dell' Informazione, Università di Padova, Italy and Dept. of Computer Science, Univ. of Helsinki, Finland. cinzia.pizzi@dei.unipd.it

biological meaning. The main problem when dealing with biosequences is posed by the intrinsic variability induced by the mutations. This translates into multiple, more or less closely resembling incarnations of the same pattern, none of which may be a perfect replica of the pattern itself. This leads to a number of candidates that grows exponentially with the number of errors admitted and burdens the discovery process unbearably. Many heuristics (for example see [3,5,7,4,6] have been adopted to cope with this explosion by pruning the search space in various ways, which makes the computation more feasible, but also the real solution to be possibly missed. Even when the limiting assumption is made that every candidate pattern has at least one exact occurrence in the textstring, the discovery process still needs to compare and score every substring against every other one. Within this framework, an alternative approach has been based on compact scoring. The idea is to build a set of classes that cover the (already reduced) search space, and to score only one representative for each class. By grouping together words that are syntactically related and score a monotonically increasing departure from expected frequency, we can assure that no word within a class is more significant or informative than the representative of that class. The application of this approach to words with mismatches (i.e., in which the exact or maximum number of errors allowed is fixed but their positions are arbitrary) was first studied in [1,2]. The expected number of occurrences, with either exactly k errors or up to k errors, was analyzed under different scenarios letting the number of errors, or the string length, or both increase. The corresponding score was proven to have a monotone behavior in all cases, except in the case of exactly k errors, when the string length is fixed and the number of errors increases. However, a lower bound for monotonicity was established, that is given by half the length of the word under analysis. In the present abstract, we further concentrate on the analysis of this case and speculate that the corresponding score exhibits a bi-tonic behavior, i.e., it either increases till the end of the word, or increases up to a certain number of errors and decreases thereupon. In order to make the present paper self-contained, the next session reports results from [2] that are relevant to the discussion. In section 3 we examine the expectation for words with mismatches when the length is fixed and the number of errors increases.

2 Preliminaries

Given an alphabet Σ , we set $w = v \cdot a$, where $w, v \in \Sigma^*$ and $a \in \Sigma$. The correction factor for a symbol a is the quantity by which the probability of a solid word w should be multiplied by if one error occurs at the position occupied by the symbol a . Under *i.i.d.* assumptions, the symbol correction factor for a is given by:

$$f_a = \frac{\sum_{s \in \Sigma \setminus \{a\}} p_s}{p_a}$$

The correction factor $C_k(w)$ for solid word w is similarly defined as the quantity by which the probability of w should be multiplied in order to take into account k errors that can occur in any of its positions.

The correction factor for w with exactly k errors can be computed in $O(k^2)$ time after a dynamic programming based on a pre-processing step that takes $O(nk)$ time (see [2] for details), or it can be computed directly in $O(k|w|)$ by applying the dynamic programming formula used in that pre-processing, and generalized by the following equation:

$$C_k(w) = C_k(v) + C_{k-1}(v)f_a. \quad (1)$$

An assumption was made that limits the skew on our probability distributions in exchange for some useful consequences. The assumption is quite reasonable for genomic as well as general applications.

Assumption 1: $p_a \leq \sum_{s \in \Sigma \setminus \{a\}} p_s$

As an immediate consequence of this assumption, we get

Property 1: $f_a \geq 1 \quad \forall a \in \Sigma$

Property 2: $C_i(w[1\dots j]) \geq 0 \quad \forall i, j$ and where $w[1\dots j]$ is the prefix of w of length j

This shows that correction factors are always non-negative, and in fact that they are positive for words longer than the number of errors k .

Property 3: $C_k(w) \geq 0 \quad \forall k, w$ and $C_k(w) \geq 1 \quad \forall w : |w| \geq k > 0$.

We discuss next some monotonicity properties of correction factors for the case (ECF) of *exactly* k errors when:

1. the word length is increased, keeping error number fixed;
2. the number of errors is increased, keeping word size fixed;
3. both word length and number of errors are increased.

Lemma 1. For $w = va$, $C_k(w) \geq C_k(v)$.

Proof. From equation 1, considering Property 1 we have $C_k(w) = C_k(v) + C_{k-1}(v)f_a \geq C_k(v) + C_{k-1}(v)$. By Property 2, $C_{k-1}(v) \geq 0$. We can conclude that $C_k(w) \geq C_k(v) + C_{k-1}(v) \geq C_k(v)$. \square

Lemma 2. For $w = va$, $C_k(w) \geq C_{k-1}(v)$.

Proof. By the argument in the previous Lemma, since $C_k(v) \geq 0$ we also have: $C_k(w) \geq C_k(v) + C_{k-1}(v) \geq C_{k-1}(v)$. \square

A counterexample will show that, in general, the correction factor is not monotonically increasing when the number of errors allowed is increased while the length of the string is kept fixed. To see this, assume that the characters of Σ have the same probability. Hence:

$$p_{s_1} = p_{s_2} = \dots = p_{s_{|\Sigma|}} = p = \frac{1}{|\Sigma|} \text{ and } f_{s_1} = f_{s_2} = \dots = f_{s_{|\Sigma|}} = f = |\Sigma| - 1$$

In this special case, for a word w we have:

$$C_k(w) = \binom{|w|}{k} f^k.$$

By the definition of $C_k(w)$ we have:

$$C_k(w) < C_{k+1}(w) \implies \binom{|w|}{k} f^k < \binom{|w|}{k+1} f^{k+1} \implies f > \frac{k+1}{|w|-k}$$

Hence $C_k(w) < C_{k+1}(w)$ holds for:

$$k < \frac{|w|f-1}{f+1}.$$

Combined with its symmetric argument, this leads to conclude that, with $\bar{k} = \lfloor \frac{|w|f-1}{f+1} \rfloor$, we have:

$$\begin{cases} C_k(w) < C_{k+1}(w) & \text{for } k \leq \bar{k} \\ C_k(w) > C_{k+1}(w) & \text{for } k > \bar{k} \end{cases}$$

Thus $C_k(w)$ is bi-tonic in this case.

The next lemma establishes an acceptable lower bound for \bar{k} , that corresponds to half the length of the string w .

Lemma 3. $C_k(w) \geq C_{k-1}(w) \quad \forall k \leq \frac{|w|}{2}.$

Proof. Let $w = w_1 w_2 \dots w_m$. The inequality holds for $k = 1$, since $C_0(w) = 1$ and:

$$C_1(w) = \sum_{i=1}^m f_i \geq \sum_{i=1}^m 1 = m \geq 1 = C_0(w).$$

The contribution of each position in this case is the correction factor of the character occupying that position. Hence we obtain a set of $\binom{m}{1} = m$ terms, which may be expressed as:

$$C_1 = (f_1, f_2, \dots, f_m).$$

For $k = 2$, we obtain a set of $\binom{m}{2} = m(m-1)/2$ terms, where each term results from the combination of the characters at two positions of w , say, w_i and w_j , and consists of the product of the corresponding correction factors $f_i f_j$. The set of contributions for $k = 2$ is given by:

$$C_2 = (f_1 f_2, f_1 f_3, \dots, f_1 f_m, f_2 f_3, \dots, f_2 f_m, \dots, f_{m-1} f_m)$$

Since $\forall i f_i \geq 1$, then $f_i f_j = f_j f_i \geq f_i \forall i, j$, so that for every term f in \mathcal{C}_1 we have at least one element \bar{f} of \mathcal{C}_2 such that $\bar{f} \geq f$. This argument propagates from one \mathcal{C} to the next for as long as the number of terms increases. But the number of terms is given by the binomial coefficients, hence our condition is preserved only for values of k up to $\frac{m}{2}$. We conclude that $\lfloor w/2 \rfloor$ is always safe as a lower bound for k . \square

A natural question at this point concerns values of k greater than $\lfloor w/2 \rfloor$. From what was seen, it is unlikely that all kinds of oscillations be possible. In the next section we deal with this specific situation.

3 Bitonicity of the expectation.

We propose here an iterative method to verify that $C_k(w)$ exhibits either a monotonic or a bi-tonic behavior when the number of errors increases and the word length is fixed.

It is convenient to refine the problem as follows. We already know that given the error number \bar{k} , for strings of length at least $2\bar{k}$, the correction factor increases when we increase the error number from 0 to \bar{k} . What we still have to verify is that for strings of shorter length, we can still guarantee monotonicity or bi-tonicity. Therefore we need to study the behavior of the correction factor for the all prefixes of w when we increase the number of errors up to exactly \bar{k} .

Towards this, we introduce a $k \times |w|$ table B to be filled as follow:

$$B[i][j] = \begin{cases} + & \text{if } C_i(w[1 \dots j]) > C_{i-1}(w[1 \dots j]) \\ - & \text{if } C_i(w[1 \dots j]) < C_{i-1}(w[1 \dots j]) \end{cases}$$

Our goal is established once we would prove that in each column of B there can be at most one sign change.

The following lemma gives a handle for our discussion:

Lemma 4. *For $w = va$, if $C_k(v) > C_{k-1}(v)$ and $C_{k-1}(v) > C_{k-2}(v)$ then $C_k(w) > C_{k-1}(w)$. Likewise, if $C_k(v) < C_{k-1}(v)$ and $C_{k-1}(v) < C_{k-2}(v)$ then $C_k(w) < C_{k-1}(w)$.*

Proof. For simplicity we discuss only the case for “>”. The proof for “<” is easily obtained by interchanging symbols.

From Equation 1 we have:

$$C_{k-1}(w) = C_{k-1}(v) + C_{k-2}(v)f_a \text{ and } C_k(w) = C_k(v) + C_{k-1}(v)f_a$$

Since $C_k(v) > C_{k-1}(v)$ and $C_{k-1}(v) > C_{k-2}(v)$, we have:

$$C_k(w) = C_k(v) + C_{k-1}(v)f_a > C_{k-1}(v) + C_{k-2}(v)f_a = C_{k-1}(w)$$

\square

As said, we hope to fill table B in such a way that in each column there is at most one sign change. After the initial set up described later, we

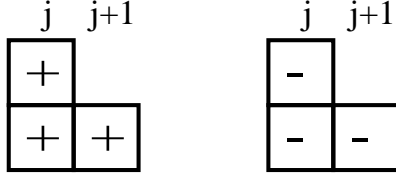


Fig. 1. Sign propagation in $B[i][j]$

will fill in the cells of the table column-wise. The lemma above tells us how to fill a cell in the next column whenever two consecutive cells in the current column have the same sign.

3.1 Initial Setup

We know from the proved lower bound that for the generic prefix $w[1 \dots j]$ the first $\lceil \frac{j}{2} \rceil$ cells are set to “+”. Thus for positions $j \geq 2k - 1$ all the columns are set to “+”, showing a monotone behavior for the corresponding prefixes. We then concentrate our attention on columns ranging from 1 to $2k - 2$.

Since at most j errors can occur in a string of length j , then the value of the correction factor for $w[1 \dots j]$ for more than j errors is 0. For such a string, the correction factor for exactly j errors is positive, so that we can set $B[j + 1][j]$ to “-” for $j = 1 \dots k - 1$. Since the value of the correction factor is still 0 for a number of errors greater than $j + 1$, we are guaranteed that in any column j no sign changes take place past row-index $j + 1$. Hence, $k - j$ more cells in column j have their value defined.

	1	2	3	4	5	6	7	8	9	10
1	+	+	+	+	+	+	+	+	+	+
2	-		+	+	+	+	+	+	+	+
3	=	-			+	+	+	+	+	+
4	=	=	-				+	+	+	+
5	=	=	=	-					+	+

Fig. 2. Initial setup of $B[i][j]$ for $k = 5$ errors

3.2 Filling the Table

By the initial setup, we have now that the cells that are still to be filled amount to $k - \lceil \frac{j}{2} \rceil - (k - j) = \lfloor \frac{j}{2} \rfloor$ for positions j , $1 \leq j \leq k$, and to $k - \lceil \frac{j}{2} \rceil$ for positions j , $k < j \leq 2k - 2$.

In particular, for $j = 1$ the column is already filled. It is seen (refer to Fig.2) that in the first column there is only one sign change. For $j = 2$ and $j = 3$ we have 1 empty cell, for $j = 4$ and $j = 5$, two empty cells, and so on: for $1 \leq t \leq \frac{k}{2}$, we have exactly t empty cells in columns $2t$ and $2t + 1$.

We start filling in the table from the second column. The top empty cell here is $B[2][2]$, and this must be filled with either a “+” or a “-”, depending on the value of the symbol correction factors. Either way, this will result in having only one sign change in the column. Moreover, if we put a “-” in $B[2][2]$ this will propagate to the cells $B[3][3]$, $B[4][4]$, and so on, along the diagonal, according to Lemma 4. This implies that in the subsequent columns one more cell has already been filled. In particular, Column 3 is completed with only one sign change, Column 4 has now 1 empty cell, etc. If, on the other hand, we set $B[2][2]$ to “+”, then no propagation occurs, because the first cell involved in the lemma has been already filled by the initial set up.

	1	2	3	4	5	6	7	8	9	10
1	+	+	+	+	+	+	+	+	+	+
2	-	-	+	+	+	+	+	+	+	+
3	=	-	-		+	+	+	+	+	+
4	=	=	-	-			+	+	+	+
5	=	=	=	-	-				+	+

	1	2	3	4	5	6	7	8	9	10
1	+	+	+	+	+	+	+	+	+	+
2	-	+	+	+	+	+	+	+	+	+
3	=	-			+	+	+	+	+	+
4	=	=	-				+	+	+	+
5	=	=	=	-					+	+

Fig. 3. Illustrating the “-” case with propagation (table of the left) and the “+” case, with no propagation (table on the right)

Following the first one of the cases considered, we can move directly to Column 4, where just one cell needs to be filled, and iterate the process.

	1	2	3	4	5	6	7	8	9	10
1	+	+	+	+	+	+	+	+	+	+
2	-	+	+	+	+	+	+	+	+	+
3	=	-	+	+	+	+	+	+	+	+
4	=	=	-				+	+	+	+
5	=	=	=	-					+	+

	1	2	3	4	5	6	7	8	9	10
1	+	+	+	+	+	+	+	+	+	+
2	-	+	+	+	+	+	+	+	+	+
3	=	-	-		+	+	+	+	+	+
4	=	=	-	-			+	+	+	+
5	=	=	=	-	-				+	+

Fig. 4. The two possible cases, both with propagation, in the further step for $B[2][2] = “+”$

In the second case, we still need to consider Column 3, however, even this case requires to set the value of one cell only. The advantage now is that whatever the sign obtained for that cell, there will be propagation to the next column (just one cell in case of a “+”, the whole diagonal in case of a “-” as shown in Fig.4). Now, when moving to Column 4 we

will have again just one cell to set up, and we can iterate the process. In Fig.5 there is an example of possible filling of the table in which at most one sign change per column occurs.




	1	2	3	4	5	6	7	8	9	10	
1	+	+	+	+	+	+	+	+	+	+	 calculated
2	-	+	+	+	+	+	+	+	+	+	 propagated
3	=	-	+	+	+	+	+	+	+	+	 init set up
4	=	=	-	+	+	+	+	+	+	+	
5	=	=	=	-	+	+	+	+	+	+	

Fig. 5. Example of a possible final configuration for Table *B*

4 Conclusions and future work.

There are cases in addition to those contemplated in the present paper that need to be addressed in order to verify that the expectation, for the range of number of errors k spanning from $|w|/2$ to $|w|$, is either monotonic or bi-tonic. This forms the subject of work in progress. It would be interesting in the future to extend the analysis of the behavior of some kind of z-scores, also reported in [2], to intervals not necessarily of constant frequency. In fact, when the counted number of occurrences increases or decreases at rate slower than the expected value, we might still be able to design compact representations for over/represented strings of the kind considered here.

References

1. Apostolico, A., Pizzi, C. “Motif Discovery by Monotone Scores” to appear in Discrete Applied Mathematics, Special Issue on Bioinformatics (2006).
2. Apostolico, A., Pizzi, C. “Monotone Scoring of Pattern with Mismatches” in Proceedings of the 4th Workshop on Algorithms in Bioinformatics (WABI 2004), Bergen, Norway, September 14 - 17, 2004 LNCS/LNBI 3240, pp 87–98.
3. Bailey, T. L., and Elkan, C. “Unsupervised learning of multiple motifs in biopolymers using expectation maximization”. Machine Learning, 21(1-2):51–80, 1995.
4. Buhler, J., Tompa, M. “Finding motifs using random projections”. Journal of Computational Biology, 9(2) 225–242, 2002.
5. Hertz, G. Z., and Stormo, G. D. Identifying dna and protein patterns with statistically significant alignments of multiple sequences. Bioinformatics, 15:563–577, 1999

6. Lawrence, C., Altschul, S., Bugoski, M., Liu, J., Neuwald, A., and Wootton, J. "Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment". *Science*, 262:208–214, 1993.
7. Stormo, G., and Hartzell, G. "Identifying protein-binding sites from unaligned dna fragments". In *Proceedings of the National Academy of Science USA*, volume 86, pages 1183–1187, 1989.