The QuickScore Library *

Mireille Régnier¹ and Edouard Dolley¹ and Magali Lescot² and Mathias Vandenbogaert¹,³

¹ Mireille.Regnier@inria.fr
INRIA Rocquencourt, Domaine de Voluceau B.P. 105, 78153 Le Chesnay - FRANCE
² LGPD-IBDM, Parc Scientifique de Luminy, CNRS Case 907 13288 Marseille Cedex 9-FRANCE
³ LaBRI - Bordeaux I, 351, cours de la Libération 33405 Talence Cedex - FRANCE

Abstract. The software library we are presenting here, QuickScore, deals with the estimation of the probabilities of rare or frequent words in genomic texts. Mathematical expressions have recently been obtained by different authors. We present a few extensions and notably address the p-value computation, in the Markov probability model and for double strand counting. A collection of functional procedures have therefore been developed for their computation. Special attention was held concerning the efficiency of the underlying algorithms and implementations. The used formula and algorithms remain tractable throughout the different statistical models. This puts forward the use of the library as a valuable benchmark for comparing the different statistical models in their range of applicability. Consequently, for the elaboration of the formula and algorithms, special attention was paid to the drawbacks of numerical instability.

Keywords: data mining, string and graph algorithms, genomics, regulatory elements, probability

1 Motivation

The huge amount of available data on the genome and the proteome is permanently increasing. It makes necessary to provide efficient tools for a prediction “in silico” of potentially interesting regions. The statistical methods are widely used [BJVU98, BLS00, LBL01, EP02] for the search of regulatory signals on an entire genome [vH03], or some part of it, or several genomes [GKM00, OLN01], or the identification of promoters [MMML02, OLN02]. They rely on a simple basic assumption: an overrepresented (or underrepresented) word, i.e. a word

* This research was partially supported by the Future and Emerging Technologies programme of the EU under contract number IST-1999-14186 (ALCOM-FT), French Bioinformatic Program and French IMPG Program.
which occurs significantly more (or less) in real sequences than in random ones, may denote a biological functionality. These words may be for example a cleavage recognition site for restriction enzymes or polyadenylation, a protein binding site such as a cis-regulatory element...

The statistical softwares combine an extraction step and an assessment step. In the first step, an algorithm detects some potential signals. Then, in the second step, one assumes a probabilistic model for the genome. According to this model, one computes, for the “candidates-motifs”, the expected frequency and compares it to the observed results. Clearly, as the number of candidates-motifs is usually very large, the speed of the assessment step is crucial.

Mathematical results have been derived recently by many authors [vHACV98, HKB02, RSW00, Reg00, ABLX00]. We briefly discuss their significance results. Our first goal is to provide new results on the so-called p-value and prove their relevance on few biological data in Section 3. Nevertheless, the proof is beyond the scope of this extended abstract. Some technical details are given in Appendix.

Our second goal is to provide a toolkit, the QuickScore library, that implement such formulae. Most functions in QuickScore are entirely written in C, and can be called by an other software. As a matter of fact, some of them are already implemented in RSA-tools [vHACV00]. The specificity of our approach is to take advantage of some combinatorial properties of words to provide tractable formulae and efficient algorithms that implement them. A special attention is paid to the numerical problems that arise in the computation of (small) real numbers. A survey and a comparison of existing softwares can be found in [Les02].

2 Significance criteria

2.1 State of the art and discussion

One can distinguish two main situations. In a first class of problems, one studies a long text—typically, a genome—and seeks for exceptional—either overrepresented or underrepresented—motifs. For example, the Chi motif in bacteria tends to be globally overrepresented along the genome, while binding sites for restriction enzymes are expected to be underrepresented [PMG00]. In a second class of problems, a large set of relatively short sequences is given, and one searches for motifs which are contained in most sequences. A typical example is the characterization of polyadenylation signals in human genes [BFW+00] or the search of cis-regulatory elements in the upstream region of orthologous or coregulated genes [DB97, BT02]. Different criteria are used, but, in any case, the over- or underrepresentation of a motif is assessed by comparing, subject to a suitable probability model, its observed number of occurrences, $N_H$, with its expected number of occurrences, $E_H$, in a random sequence (or a random set of sequences). Different background probability models are discussed in [HKB02]. In this paper, we assume that the model is Markovian, with an order ranging from 1 to 3, or Bernoulli. We discuss below the validity domain of commonly used criteria.
1. A widely used selection criteria is the Z-score:

\[ Z(H) = \frac{N_H - E_H}{\sqrt{V_H}}. \]

2. A more sensitive criterion is the so-called p-value, e.g., the probability to find that observed number \( N_H \) of occurrences for the signal under study:

\[ p - \text{value}(H) = P(\text{Number of occurrences} \geq \text{observed number}). \]

**Z-score paradox** The theoretical support for the Z-score use is the well known result derived in [BK93]: the number of occurrences converge in distribution to a normal law. Similar and extended results can be found in [BK93, RS97, NSF99, Régo00, RSW00]. Intuitively, this convergence result means that the Z-score of a usual word should be 0 and, typically, a high positive (respectively small negative) Z-score detects the overrepresentation (respectively the underrepresentation). Still, this significance relies on an asymptotic property. As a first consequence, as it was observed by J. van Helden, it is irrelevant for the study of short sequences. This problem is addressed below. Second, even for long sequences, it does not provide valuable information when the motifs are really exceptional. The p-values order may be totally different from the Z-score orders. This fact was illustrated for overrepresented words in [DRV01]. For underrepresented words, the order of strongly negative Z-scores is weakly related to the avoidance. Mathematically, when the observed number \( N_H \) is much smaller than the expectation \( E(H) \), then, the Z-score is upper bounded by \( \sqrt{E(H)} \). (with the current approximation \( E(H) = V_H \)). The higher the probability of the words, the higher the potential Z-scores! This **Z-score paradox** is illustrated in Table 1 in Section 3 for palindromic words that are avoided on the bacterial genomes. For a given genome, the palindromic sites associated to the restriction enzymes of the genome are avoided [PMG00]. Nevertheless, the p-values order is significantly different from the Z-score orders.

As a consequence, one cannot precisely compare two words with different expectations on the basis of their Z-scores. Nevertheless, QuickScore provides an efficient computation of the Z-score for some important signal structures in the cases where it is a significant filter. General formulae for the mean and variance of a set of words have been derived by various authors [RSW00]. Namely:

\[ \text{Exp}(\mathcal{H}) = n \sum_{H \in \mathcal{H}} P(H) \]

\[ \text{Var}(\mathcal{H}) = n \sum_{H \in \mathcal{H}} P(H) \times \left[ 1 + (2m - 1) \sum_{H \in \mathcal{H}} P(H) + C(\mathcal{H}) \right] \]

where \( C(\mathcal{H}) \) depends on the word overlaps. This algorithmic problem of the computation of \( \sum_{H \in \mathcal{H}} P(H) \) and \( C(\mathcal{H}) \) is detailed in Section 4. Interestingly, the correlation factor \( C(\mathcal{H}) \) is also needed for small sequences.
\textbf{p-values}.

We will address the problem of the precise and fast computation of the \textit{p}-value for long and short sequences.

\textit{Long sequences} When a high Z-score indicates that a word is exceptional, one wants to assess the significance of the signal by a computation of its \textit{p}-value. Intuitively, the largest is the difference (or the ratio) between the observation, \( O \), and the expectation, \( E \), the smallest is the probability of this event. As a matter of fact, it is shown in [RS97, NSF99] that the \textit{p}-value is exponentially decreasing with \( E - O \). The best program addressing this is \texttt{Excep} described in [KBCC00]; it is precise and numerically stable in a large domain of parameters, but it is computationally expensive. Our recent large deviation results in teh Bernoulli model [DRV01, RD03] allow for a precise and cheap computation of the \textit{p}-value. More precisely:

\[
\frac{\log[p - \text{value}]}{n} \sim I\left(\frac{E - O}{n}\right)
\]

where \( I \) is a function that depends on the possible self-overlaps of \( H \) and the probability distribution through a polynomial equation. Hence, this \textit{p}-value computation reduces to the numerical solution of a polynomial equation. This is implemented in Maple in QuickScore and available on our web site. Clearly, such a solution is much faster than the computation by induction of \texttt{Excep} [KBCC00] or \texttt{GDon} [Nue01], which has an exponential complexity, in the Markov case. Moreover, this approximation is very tight. It is as precise as the software \texttt{Excep}, when this software is not stalled by its high complexity.

The formulae, formally proved in [RD03] extend for a Markov model. The additional cost comes from the computation of a Markovian contribution that reduces to the division of a transition matrix [RS97]. Again, one only need to solve a (slightly modified) polynomial equation.

This improvement is relevant. Table 1 shows that the order of Bernoulli and Poisson \textit{p}-values may be totally different.

\textit{Small sequences} A very classical and useful problem [BFW+00, BT01] is as follows. Given a set of \( L \) small sequences -typically the upstream sequences of genes-, extract a signal that is common to these sequences. In a statistical approach, one computes first the probability \( p \) that a given word (or set of words) occurs at least once in a given sequence. Although the parameter \( p \) computation is much trickier than the mean/variance computation, this evaluation is made possible due to the small size of the sequences [RS97]. Additonnally, a cheap and precise approximation, that is valid for self-overlapping words is presented in [Rég02].

\[
p \sim 1 - e^{-n \frac{p(H)}{E(H)}}
\]

This generalized approximation is very tight. Still, the approximation is always a lower bound for \( p \); hence, \( p \) is slightly underestimated and this leads to an
underestimation of $L_p$, the expected number of matching sequences. This fact was observed by J. van Helden [vH02] from simulations on random sequences.

Then, one evaluates the probability $P_L(k)$ that it occurs in $k$ of these $L$ sequences. The smaller this $p$-value, the more exceptional, and significant, is the word. We point out here that this simply is a Bernoulli process with parameters $(L, k, p)$:

$$P_L(k) = \sum_{K \geq k} \binom{L}{K} p^K (1-p)^{L-K}.$$  

Hence, it steadily derives from Stirling’s formula (1750) that $P_L(k) = e^{-L \cdot I(\hat{p})}$ with

$$I(a) = a \log \frac{a}{p} + (1-a) \log \frac{1-a}{1-p}.$$ 

(2)

This approximation is very tight. Not only its computation is very fast, but the classical problems of computation on the real numbers make it more precise, numerically, than the computation of the exact formula with a common precision on the real numbers. This improvement has been added to RSA-tools, and will be soon available on QuickScore. The double strand formulae above also apply for short sequences.

2.2 Double strand counting

Counting motifs is usually performed on one strand only. However, a binding site may occur on either one of the two strands, and, usually, there is not enough information to allow to chose the correct strand. The statistics of the counting on two strands is hard, due to the fact that the two strands are not independent: indeed, they are complementary. Still, we simply observe that the number of occurrences of a word $H$ on two strands is equal to the number of occurrences of $H$ and its complement $\hat{H}$ on any of the two strands. For a set of motifs, this leads to a weighted counting scheme described in [RLM00].

The general formulae for the mean and variance [Rég00] of the occurrences of a set of words greatly simplify for the set $\{H, \hat{H}\}$. The reason for that is that all results depend on the inverse of a matrix, $D$, the dimension of which being exactly the size of the set, which is 2 for $\{H, \hat{H}\}$. Hence, we applied the classical inversion formulae and realized a Maple implementation. We tested them to extract the i-box from promoter sequences of plant regulons. This approach is more sensitive than RSA-tools, that uses a very rough approximation to take into account the two strand dependency. Again, the $p$-value computation reduces to the solution of a polynomial equation, that depends on the determinant of this matrix.

3 Results in biology

3.1 Restriction-Modification Systems

Z-score paradox A palindrome is a symmetric word: two letters located at the same distance from the central position of the word are complementary according
to the Watson-Crick rule: $A \leftrightarrow T, C \leftrightarrow G$. For a given genome, the palindromic sites associated to the restriction enzymes of the genome are avoided [PMG00]. Nevertheless, the table below shows that the $p$-values order is different from the Z-score orders.

<table>
<thead>
<tr>
<th>Hexamer</th>
<th>Occ</th>
<th>Z-score</th>
<th>$Z_{\text{max}}$</th>
<th>$P$-val (B)</th>
<th>$P$-val (M)</th>
<th>$M$-Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCCGGC</td>
<td>94</td>
<td>-41.9</td>
<td>-44.1</td>
<td>1.594571143e-3</td>
<td>4.338655443e-3</td>
<td>1</td>
</tr>
<tr>
<td>GCAAGT</td>
<td>588</td>
<td>-39.9</td>
<td>-51.4</td>
<td>0.37e-4</td>
<td>15.251409646e-3</td>
<td>2</td>
</tr>
<tr>
<td>GCGGGC</td>
<td>293</td>
<td>-37.4</td>
<td>-41.9</td>
<td>1133046834e-3</td>
<td>10.235816966e-3</td>
<td>3</td>
</tr>
<tr>
<td>CGCGGC</td>
<td>588</td>
<td>-35.1</td>
<td>-56.9</td>
<td>1564951908e-3</td>
<td>1849237432e-3</td>
<td>5</td>
</tr>
<tr>
<td>GCCGGG</td>
<td>68</td>
<td>-30.7</td>
<td>-32.7</td>
<td>2113074684e-3</td>
<td>1.763897841e-3</td>
<td>6</td>
</tr>
<tr>
<td>TTGCAA</td>
<td>1024</td>
<td>-30.2</td>
<td>-50.4</td>
<td>1.53069438e-6</td>
<td>2.212089661e-3</td>
<td>7</td>
</tr>
<tr>
<td>ATGATC</td>
<td>43</td>
<td>-28.0</td>
<td>-46.1</td>
<td>7.06686859e-3</td>
<td>1.695910354e-3</td>
<td>9</td>
</tr>
<tr>
<td>GCCGGC</td>
<td>567</td>
<td>-26.2</td>
<td>-50.4</td>
<td>3624131701e-4</td>
<td>14.98027091e-4</td>
<td>10</td>
</tr>
<tr>
<td>GCTAGC</td>
<td>157</td>
<td>-26.1</td>
<td>-31.1</td>
<td>1.495565720e-3</td>
<td>7.113312547e-3</td>
<td>8</td>
</tr>
<tr>
<td>CAAAAG</td>
<td>185</td>
<td>-25.7</td>
<td>-46.8</td>
<td>1.24717946e-4</td>
<td>9.04588694e-4</td>
<td>12</td>
</tr>
<tr>
<td>GAGCTC</td>
<td>52</td>
<td>-23.9</td>
<td>-29.1</td>
<td>1.524757568e-3</td>
<td>9.436368542e-4</td>
<td>11</td>
</tr>
<tr>
<td>GTGCAC</td>
<td>575</td>
<td>-23.1</td>
<td>-38.1</td>
<td>4006964500e-4</td>
<td>2.753139868e-4</td>
<td>14</td>
</tr>
<tr>
<td>TGGCAA</td>
<td>630</td>
<td>-23.0</td>
<td>-39.1</td>
<td>3222046366e-4</td>
<td>7.738769289e-4</td>
<td>15</td>
</tr>
<tr>
<td>GAATTC</td>
<td>645</td>
<td>-21.9</td>
<td>-38.6</td>
<td>2360605321e-4</td>
<td>1.664817246e-4</td>
<td>19</td>
</tr>
<tr>
<td>AAGCTT</td>
<td>556</td>
<td>-21.9</td>
<td>-36.9</td>
<td>3519853042e-4</td>
<td>1.767111153e-4</td>
<td>18</td>
</tr>
<tr>
<td>CTGAGG</td>
<td>177</td>
<td>-21.3</td>
<td>-27.7</td>
<td>1.418943480e-3</td>
<td>1.721997639e-4</td>
<td>17</td>
</tr>
<tr>
<td>CCTAGG</td>
<td>16</td>
<td>-21.2</td>
<td>-21.9</td>
<td>2.338836346e-3</td>
<td>1.653976280e-4</td>
<td>13</td>
</tr>
<tr>
<td>ACATCT</td>
<td>477</td>
<td>-20.4</td>
<td>-34.3</td>
<td>4.80689925e-4</td>
<td>1.581277978e-4</td>
<td>20</td>
</tr>
<tr>
<td>TCTAGA</td>
<td>87</td>
<td>-19.8</td>
<td>-21.6</td>
<td>2.000305392e-3</td>
<td>1.718962476e-4</td>
<td>16</td>
</tr>
<tr>
<td>TTGGAA</td>
<td>800</td>
<td>-19.4</td>
<td>-39.6</td>
<td>9.95721002e-4</td>
<td>4.903391920e-4</td>
<td>22</td>
</tr>
<tr>
<td>CCATGG</td>
<td>612</td>
<td>-19.4</td>
<td>-36.2</td>
<td>3.46704707e-4</td>
<td>1.608649839e-4</td>
<td>21</td>
</tr>
<tr>
<td>CGCGCG</td>
<td>2127</td>
<td>-118.7</td>
<td>-56.4</td>
<td>.</td>
<td>3.641666775e-4</td>
<td>23</td>
</tr>
</tbody>
</table>

**Table 1.** Table of avoided palindromic hexanucleotides in *E. Coli*

Column 2 yields the number of occurrences in *E. Coli.*

The first two binding sites that exchange their ranks are GCTAGC and ATGCA for enzymes *NheI* and *AvaIII*; (8, 10) changes to (9, 8). When a Z-score is closed to its maximum, then its rank according to the $P$-value (the $P$-rank in column 8) greatly improves. For instance, CCTAGG (enzyme *AvaII*) goes from rank 18 up to rank 13, and TCTAGA (enzyme *XbaI*) goes from rank 20 up to rank 16. For similar Z-scores, the $p$-value, in column 7, reverses such order when the maximal Z-order ar different. This is the case, for example, the three binding sites TCTAGA, TTGGAA and CCATGG for enzymes *XbaI*, *AvaII* and *Neol*. It is interesting to notice, that the last hexamer, CGCGCG, keeps its rank, with a sensible difference in $p$-value (a 0.13e-4
difference in the logarithms represents a $10^{-26}$ factor for the $p$-value), is not a binding site. More detailed results on different organisms are available at (http://pauillac.inria.fr/algo/vandenbogaert/RE/CompareModels/ecoli_MaxM_Plane_K1.html).

3.2 Plants datasets

Three datasets based on dicot plant sequences were collected using the query system of PlantCARE database (http://oberon.rug.ac.be:8080/PlantCARE/index.html) described in [LDT+ 02] and the sequence retrieval tool of INCLUSive [TMS+02] available at (http://www.esat.kuleuven.ac.be/ dna/Bio1/Software.html). They are sets of upstream sequences of genes regulated by a same transcription factor (also called regulon). The size of each sequence is up to 800 bp. In pattern discovery the redundancy of sequences in the data set is an important problem. Related sequences could strongly bias the probabilistic calculation by counting twice all the oligonucleotides of the same sequence. The program Purge_sequence from Stefan Kurtz (installed at the web site of RSA-tools http://rsat.ulb.ac.be/rsat/) allowing to remove the repeated regions in the sequences was used first. Once the datasets purged, the program Oligo-analysis [vHACV98] from RSA-tools was ran on the different plant regulons. The expected frequency for each word was calculated on basis of observed subword frequencies, according to a Markov chain model [vHOPO01]. QuickScore $p$-value was computed for these top scoring words.

The G-box regulon contains 31 promoter sequences having at least one binding site per sequence recognized by the G-box Binding Protein (GBP). The G-box (CACGTG) is a very well conserved regulatory element [DC90]. The description of G-box dataset is available at (http://www.psb.rug.ac.be/bioinformatics/lescot/Datasets/ListG-boxes.html) or in [TLM+01].

It turns out from these results that the Bernoulli model is good enough to extract the G-box, which is a very strong signal. The RSA-heuristics and QuickScore both find it at the top. It is worth noticing that the i-box, that is also in this data set, appears with QuickScore as the second non-TATA-like signal. In the seventh position, the palindromic motif GCATGC (found only at the 28th position with Oligo-analysis) could be part of a RY-element (CATGCATG). This cis-regulatory element is known to be involved in seed-specific regulation (data from PlantCare database). Recently the co-occurrence of the g-box and RY-elements have been identified in the promoter of the phas gene in Phaseolus vulgaris [CBH03].

The second regulon was built based on 17 sequences containing an I-box (GATAAG). This regulatory element, compared to the G-box, is considerably less conserved (more degenerated).

Table 3 shows the results for a Markov model of order 1, using QuickSoer and RSA-tools. The known regulatory sites are found among the top seven words (against the top ten words using Oligoanalysis). The word AGATAA (containing the i-box) went up in the list from the 10th to the 4th place, GATAAG (also containing the i-box) goes down to 6-th rank from its 5-th rank and the G-box
(CACGTG) is now in the 5th position (against the 9th). Among the top seven words, the TATA-box is still in the first position and the second and third words are TATA-like sites. The next signal, AACAG, has a significantly smaller value (0.0004 yields a $10^{-24}$ factor). These results are consistent with what is known about the sequences used to make the dataset. For instance, G-box regulatory sites have been described in some of the sequences of the i-box dataset. Moreover as the sequences were extracted upstream of the transcription site of the gene, we did expect that these sequence contain a TATA-box.

A set of 17 sequences was constituted with a collection of prolamino-box which is the binding site of the prolamino-box binding factor (PBF). The conserved motif is TGTAAG.

Although GTAAAG leaves its 1st rank to the irrelevant TCCAG (3rd with

<table>
<thead>
<tr>
<th>H1</th>
<th>H2</th>
<th>P-value</th>
<th>P-rank</th>
<th>RSA-rank</th>
<th>RSA-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>caagtc</td>
<td>caagtc</td>
<td>0.0005286712</td>
<td>1</td>
<td>1</td>
<td>18.97</td>
</tr>
<tr>
<td>atatat</td>
<td>atatat</td>
<td>0.003490670579</td>
<td>2</td>
<td>2</td>
<td>14.51</td>
</tr>
<tr>
<td>tatata</td>
<td>tatata</td>
<td>0.002909704076</td>
<td>3</td>
<td>3</td>
<td>13.35</td>
</tr>
<tr>
<td>agcggc</td>
<td>caagtc</td>
<td>0.00030184508</td>
<td>16</td>
<td>7</td>
<td>5.63</td>
</tr>
<tr>
<td>aaaaaa</td>
<td>tttttt</td>
<td>0.001950411598</td>
<td>4</td>
<td>5</td>
<td>11.72</td>
</tr>
<tr>
<td>agagag</td>
<td>ctctct</td>
<td>0.000684331484</td>
<td>14</td>
<td>6</td>
<td>8.93</td>
</tr>
<tr>
<td>acacgt</td>
<td>acagtc</td>
<td>0.0006030184508</td>
<td>16</td>
<td>7</td>
<td>5.63</td>
</tr>
<tr>
<td>gacaga</td>
<td>ctctctc</td>
<td>0.0004850796689</td>
<td>18</td>
<td>8</td>
<td>5.66</td>
</tr>
<tr>
<td>cgtgac</td>
<td>gacagc</td>
<td>0.0009731707396</td>
<td>9</td>
<td>9</td>
<td>5.50</td>
</tr>
<tr>
<td>atgcca</td>
<td>tgcagc</td>
<td>0.0007546907283</td>
<td>12</td>
<td>10</td>
<td>5.22</td>
</tr>
<tr>
<td>ctctct</td>
<td>gacagc</td>
<td>0.0008981282100</td>
<td>10</td>
<td>11</td>
<td>4.02</td>
</tr>
<tr>
<td>tataata</td>
<td>tatttt</td>
<td>0.000044411667</td>
<td>26</td>
<td>12</td>
<td>2.67</td>
</tr>
<tr>
<td>agaaaa</td>
<td>ttcttc</td>
<td>0.0000101527606</td>
<td>8</td>
<td>13</td>
<td>2.32</td>
</tr>
<tr>
<td>agagag</td>
<td>ttcttc</td>
<td>0.0007301250559</td>
<td>13</td>
<td>14</td>
<td>2.11</td>
</tr>
<tr>
<td>aagaga</td>
<td>ttcttc</td>
<td>0.00061953492872</td>
<td>15</td>
<td>15</td>
<td>1.70</td>
</tr>
<tr>
<td>tgtgca</td>
<td>tgccac</td>
<td>0.0005156594314</td>
<td>17</td>
<td>16</td>
<td>1.68</td>
</tr>
<tr>
<td>aagaga</td>
<td>ttcttc</td>
<td>0.000763203980</td>
<td>11</td>
<td>17</td>
<td>0.98</td>
</tr>
<tr>
<td>agaggg</td>
<td>cctcct</td>
<td>0.0005809071942</td>
<td>20</td>
<td>18</td>
<td>0.88</td>
</tr>
<tr>
<td>atatta</td>
<td>taatata</td>
<td>0.000031865340</td>
<td>27</td>
<td>19</td>
<td>0.86</td>
</tr>
<tr>
<td>cccttc</td>
<td>gacagc</td>
<td>0.0002274952899</td>
<td>23</td>
<td>20</td>
<td>0.74</td>
</tr>
<tr>
<td>tataata</td>
<td>tatttt</td>
<td>0.0000302504021</td>
<td>28</td>
<td>21</td>
<td>0.62</td>
</tr>
<tr>
<td>agagag</td>
<td>ttcttc</td>
<td>0.0002925486336</td>
<td>22</td>
<td>22</td>
<td>0.55</td>
</tr>
<tr>
<td>agagag</td>
<td>cctcct</td>
<td>0.0001810895985</td>
<td>25</td>
<td>23</td>
<td>0.50</td>
</tr>
<tr>
<td>ctcttc</td>
<td>gataag</td>
<td>0.0001755126172</td>
<td>5</td>
<td>24</td>
<td>0.48</td>
</tr>
<tr>
<td>aataata</td>
<td>ttattt</td>
<td>0.000467987509</td>
<td>19</td>
<td>25</td>
<td>0.38</td>
</tr>
<tr>
<td>cgtgct</td>
<td>gacagc</td>
<td>0.00003118252875</td>
<td>21</td>
<td>26</td>
<td>0.09</td>
</tr>
<tr>
<td>gagaaga</td>
<td>ttcttc</td>
<td>0.0002137461048</td>
<td>24</td>
<td>27</td>
<td>0.08</td>
</tr>
<tr>
<td>gcagtc</td>
<td>gcagtc</td>
<td>0.001133615113</td>
<td>7</td>
<td>28</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Table 2. G-box: RSA-tools and QuickScore under the Bernoulli model.
### Table 3. I-box: RSA-tools and QuickScore.

Column 3 yields the p-value from QuickScore, Columns 5 and 6 yield the rank and the score from RSA-tools.

<table>
<thead>
<tr>
<th>H1</th>
<th>H2</th>
<th>P-value</th>
<th>P-rank</th>
<th>RSA-rank</th>
<th>RSA-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>tattt</td>
<td>tattt</td>
<td>0.01845856718</td>
<td>1</td>
<td>1</td>
<td>13.92</td>
</tr>
<tr>
<td>tattt</td>
<td>tattt</td>
<td>0.0000191179417</td>
<td>17</td>
<td>2</td>
<td>9.26</td>
</tr>
<tr>
<td>acatca</td>
<td>acatca</td>
<td>0.01629024226</td>
<td>2</td>
<td>3</td>
<td>8.68</td>
</tr>
<tr>
<td>acacac</td>
<td>acacac</td>
<td>0.0005641676079</td>
<td>9</td>
<td>4</td>
<td>4.42</td>
</tr>
<tr>
<td>accttac</td>
<td>gataac</td>
<td>0.002014702806</td>
<td>6</td>
<td>5</td>
<td>3.29</td>
</tr>
<tr>
<td>acgcggc</td>
<td>acgctt</td>
<td>0.004544684399</td>
<td>10</td>
<td>6</td>
<td>2.35</td>
</tr>
<tr>
<td>agacct</td>
<td>acacac</td>
<td>0.001634645437</td>
<td>7</td>
<td>7</td>
<td>1.95</td>
</tr>
<tr>
<td>agataa</td>
<td>tatctt</td>
<td>0.002102778614</td>
<td>4</td>
<td>10</td>
<td>1.49</td>
</tr>
<tr>
<td>agaatata</td>
<td>tatattt</td>
<td>0.002138020588</td>
<td>3</td>
<td>11</td>
<td>1.15</td>
</tr>
<tr>
<td>gcgtgcccacgc</td>
<td>0.01220101817</td>
<td>8</td>
<td>13</td>
<td>0.22</td>
<td></td>
</tr>
<tr>
<td>acaagc</td>
<td>ctctgc</td>
<td>0.001492306044</td>
<td>13</td>
<td>14</td>
<td>0.21</td>
</tr>
<tr>
<td>acacac</td>
<td>acacac</td>
<td>0.002176596548</td>
<td>11</td>
<td>15</td>
<td>0.11</td>
</tr>
<tr>
<td>ctctct</td>
<td>ggcacg</td>
<td>0.001151632612</td>
<td>14</td>
<td>16</td>
<td>0.10</td>
</tr>
<tr>
<td>gtcgggtc</td>
<td>gcaagc</td>
<td>0.001220101817</td>
<td>8</td>
<td>12</td>
<td>0.80</td>
</tr>
</tbody>
</table>

### Table 4. prolamin-box: RSA-tools and QuickScore under the Bernoulli model.

<table>
<thead>
<tr>
<th>H1</th>
<th>H2</th>
<th>P-value</th>
<th>P-rank</th>
<th>RSA-rank</th>
<th>RSA-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cattac</td>
<td>gtaag</td>
<td>0.00139835704</td>
<td>2</td>
<td>1</td>
<td>3.69</td>
</tr>
<tr>
<td>agatga</td>
<td>tctctt</td>
<td>0.001315073051</td>
<td>3</td>
<td>2</td>
<td>2.57</td>
</tr>
<tr>
<td>tcacac</td>
<td>ttctgga</td>
<td>0.001520395849</td>
<td>1</td>
<td>3</td>
<td>2.29</td>
</tr>
<tr>
<td>tataaattata</td>
<td>0.000372483347</td>
<td>16</td>
<td>4</td>
<td>2.04</td>
<td></td>
</tr>
<tr>
<td>gactct</td>
<td>gactct</td>
<td>0.001001893629</td>
<td>10</td>
<td>5</td>
<td>1.51</td>
</tr>
<tr>
<td>gactct</td>
<td>gactct</td>
<td>0.001001893629</td>
<td>10</td>
<td>6</td>
<td>1.51</td>
</tr>
<tr>
<td>gttaaatatta</td>
<td>0.001212588854</td>
<td>4</td>
<td>7</td>
<td>1.32</td>
<td></td>
</tr>
<tr>
<td>ccaaaaatttggga</td>
<td>0.001142295768</td>
<td>5</td>
<td>8</td>
<td>1.31</td>
<td></td>
</tr>
<tr>
<td>agataga</td>
<td>tctctt</td>
<td>0.000353385162</td>
<td>11</td>
<td>9</td>
<td>1.05</td>
</tr>
<tr>
<td>cttgta</td>
<td>tcaacg</td>
<td>0.000728024223</td>
<td>12</td>
<td>10</td>
<td>1.05</td>
</tr>
<tr>
<td>agtcatgac</td>
<td>acatgac</td>
<td>0.001074979528</td>
<td>6</td>
<td>11</td>
<td>0.57</td>
</tr>
<tr>
<td>aaaaagccttttt</td>
<td>0.000712743838</td>
<td>13</td>
<td>12</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td>acatgatagatgat</td>
<td>0.000571133009</td>
<td>15</td>
<td>13</td>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>aaaggtaccttt</td>
<td>0.000646761719</td>
<td>14</td>
<td>14</td>
<td>0.17</td>
<td></td>
</tr>
<tr>
<td>caggggacgacctag</td>
<td>0.001065492336</td>
<td>8</td>
<td>15</td>
<td>0.17</td>
<td></td>
</tr>
<tr>
<td>catgacgcgtgac</td>
<td>0.00100876948</td>
<td>7</td>
<td>16</td>
<td>0.03</td>
<td></td>
</tr>
</tbody>
</table>
RSA), the signal TGTAAA goes up to 4th rank from 7th. A further study on heptamers should improve this slight drawback. More results are available at http://pauillac.inria.fr/algo/regnier/plantes.html.

4 Algorithms and implementation

4.1 Pattern consensus

It turns out that, due to the possible occurrence of mutations, the signal can be degenerated and still remain functional. A large variation of patterns is observed in the sequences of experimentally confirmed sites. Formally, for a given motif $H$, a *neighbourhood* of $H$ is a set $\mathcal{H}$ of patterns that are derived from $H$ by a finite number of substitutions. The (naive) complexity of the computation is $O(|\mathcal{H}|)$ for the expectation $\sum_{H \in \mathcal{H}} P(H)$ and $O(|\mathcal{H}|^2)$ for the variance, that depends on the correlation factor. Additionally, each probability is very small, and this may introduce some numerical instabilities. The formulae and algorithms elaborated below drastically reduce this complexity.

*Expectation(s)* One considers in turn:

1. Procedures *probaNeighKB, probaNeighKM*.

The procedures *probaNeighKB* and *probaNeighKM* compute the expectation of a $k$-neighbourhood of a motif. $k$ is the number of substitutions allowed (ranging from 0 to 10 in our implementation). In the Bernoulli model, a simple application of the theory of generating functions implies that the mean is a function of the parameters $\sigma_k = \sum_{i=1}^{\left|H\right|} \frac{1}{P(H)}$, where $H_i$ is the $i$-th character of $H$. This function was derived with the Maple symbolic computation system [RDLV02]. Calculating $\sigma_k$ is a simple substitution of actual values in this function, and proves to be fast and numerically stable.

*Example 1.* For a given word "AGGCAAAT", with letter frequencies given as $[P(A), P(C), P(G), P(T)] = [0.29, 0.24, 0.26, 0.21]$, and when 1 error allowed to occur, the resulting expectation is 0.000540.

The procedure *probaNeighKM* is used for the Markovian case, currently of first order. At step $l$, $1 \leq l \leq k$, the expectations of $m$ subsets of the $l$-neighbourhood are computed: each subset contains the $l$-neighbours with the leftmost error at position $i$, with $1 \leq i \leq m$. The key improvement is the precomputation and the memorization of the cumulated expectations of $H$-suffixes of $H$-occurrences. The contribution of any possible substitutions at any position is computed in a vector-like data structure.

These values are iteratively updated, starting with the case where no error is allowed to occur, and ending with $kmax$ (the number of maximum allowed errors). With every update, each contribution to the expectation is being added up.

The cost of each step is $m^2$, which leads to an overall complexity of $O(mk^2)$ for this dynamic programming algorithm. The cost of a naive computation,
that sums the probabilities of all the patterns in the neighbourhood, is equal
to the size of the neighbourhood, e.g. \( m^k \).

2 Procedure position_specific.probaNeighKB, probaB_Iupac, probaM_Iupac.

Allowing any error at any position introduces too much noise in the statistical
measure. Nevertheless, the impact of the error-range can be controlled. These
procedures are specialized from the above procedures for patterns where
mistakes are allowed on specific positions. A special case comes from the use
of IUPAC ambiguity code. These position specific versions of the expectation
reduces the number of false positive candidates that are dealt with when all
possible substitutions are considered. Additional arguments must be given to
the procedure, including the position and the kind of substitution we want
to allow there. Therefore, efficient IUPAC code definition and coding has
been included in our procedures.

3 Procedure PosPal

The study of the palindromes provides explanations in different biological
processes (see the RMS above). Taking into account the particular structure
of palindromes, we have specialised the above functionalities for palindromes,
in the Markov model and for the IUPAC code.

Z-scores One computes a FastZScore or, when necessary, a FullZScore.

It is common to approximate the variance \( \text{Var}(H) = E(N_H^2) - E(H)^2 \) by the
expectation \( E(H) \) [HRSC00, Nic01]. According to the discussion in [RLM00] and
the simulations by J. Van Helden [vH02], this approximation is correct and only
needs a small correction, when the word(s) has a few overlaps. Hence, FastZScore
computes \( Z = \frac{\text{Obs}(H) - E(H)}{\sqrt{E(H)^2}} \).

For some words, typically the polyAs, one cannot approximate \( \text{var}(H) \) by
\( E(H) \). Interestingly, when the FastZScore is small enough, a recomputation ap-
ppears unnecessary, as the relative error is bounded. Otherwise, a more precise
computation of the variance is necessary, for a correct evaluation of the Z-score.
FullZScore calls the SmartVar algorithm. This procedure calculates the variance
contribution for a neighbour of a motif, through an efficient computation of the
correlation factor \( C(H) \). It relies on the observation that the overlapping posi-
tions of any two words in a neighbourhood of \( H \) are the self-overlapping positions
of \( H \), with some exceptions. E.g. the cumulated expectations of \( H \)-suffixes of
\( H \)-occurrences are, again, precomputed. SmartVar was succesfully used for in
silico prediction for the Drosophila [PML+02].

4.2 Web site

The current version of our Web site is available at the address
http://algo.inria.fr/dolley/QuickScore.html.

The main part of the site is a form which allows to call the functionalities dis-
cussed above. At the first step, the user enters useful information on the pattern
to be studied, and the errors that are allowed. At the second step, the background
probability model is chosen. The markovian matrices for various organisms have
been percomputed using genomic sequences found on the site of GeneBank and the stationary probabilities were estimated from these matrices. A similar approach was used by Jacques van Helden for his software RSA tools and a similar database is maintained at the address \texttt{http://rsat.ulb.ac.be/rsat/}. Hence, the user can chose any of these organisms, He can also enter parameters of his own, so that he can use our functionalities for an organism that is not supported by our database.

5 Conclusion

These functions will be extended, using new mathematical results. Notably, QuickScore will take into account dyads [EP02] or structured motifs [MS00]; these are consensus patterns separated by a fixed or bounded number of unspecified positions.

6 Acknowledgments

We are grateful to Jacques van Helden and Ludovic Meunier for many fruitful discussions.

References


7 Appendix

Given two words H and F, the correlation set \( A_{H,F} \) is the set of proper suffixes \( w \) of F such that F is a suffix of H \( \cdot w \). For example, when H = ACATACA and F = ACAGT, then \( A_{H,F} = \{ GT, CAGT \} \).

The correlation polynomial is \( A_{H,F}(z) = \sum_{w \in A_{H,F}} p(w) z^{|w|} \) where \( |w| \) is the length of \( w \), and \( p(w) \) is either the probability of \( w \) (Bernoulli case) or the probability of \( w \), conditioned by an occurrence of H left of \( w \) (Markov case).

The correlation factor of a set \( \mathcal{H} \) is:

\[
C(\mathcal{H}) = \sum_{H,F \in \mathcal{H}} A_{H,F}(1)
\]

Assume the text is randomly generated by a stationary Markovian process of order 1 on an alphabet of size V. Let \( \mathbf{P} \) be its transition matrix and \( \pi \) be the vector of its stationary probabilities. Define \( \Pi \) as the \( V \times V \) matrix with \( V \) rows equal to \( \pi \) and let \( \mathbf{G} \) be:

\[
G = (\mathbf{P} - \Pi)(\mathbf{I} - (\mathbf{P} - \Pi)\mathbf{v})^{-1}
\]

Given two words H and F, the markovian contribution is

\[
m_{H,F} = \frac{G_{i(\Pi),j(F)}}{\pi[f(F)]}
\]

where \( l(H) \) and \( f(F) \) are the last and the first character of H and F, respectively.

All counting results depend [Rég00] on the set \( \{ A_{H,F}(z), P(H), m_{H,F} \}_{H,F \in \mathcal{H}} \). One needs define the polynomials

\[
D_{H,F}(z) = (1 - z)A_{H,F}(z) + P(F)z^m + m_{H,F}
\]

One define \( \mathbf{D} \) as the 2 \( \times \) 2 matrix

\[
\begin{pmatrix}
D_{H,H}(z), & D_{H,F}(z) \\
D_{F,H}(z), & D_{F,F}(z)
\end{pmatrix}
\]

**Theorem 1.** Let H be a given pattern, and \( P(H) \) be its probability of occurrence. Let \( a \) be a real positive number such that \( 0 < a < 1 \) and \( a \neq P(H) \). Let \( z_a \) be the largest real positive root satisfying \( 0 < z_a < 1 \) of

\[
D_{H,H}(z) - (1 + (a - 1)z)D_{H,H}(z) - az(1 - z)D_{H,H}(z) = 0,
\]

if \( a > P(H) \), or the smallest real positive solution that satisfies \( 1 < z_a \), if \( a < P(H) \). Then, the probability that H occurs more than \( n_a \) times in a sequence of length \( n \) satisfies:

\[
-\log \frac{\text{Prob}(N_H \geq n_a)}{n} \sim I(a)
\]
where

\[ I(a) = a \ln \left( \frac{D_{H,H}(z_a)}{D_{H,H}(z_a) + z_a - 1} \right) + \ln z_a \] (5)