RESEARCH

Signal filtering for evaluating the similarity between DNA sequences

Helena C.G. Leitão[1*], Rafael F.V. Saracchini[2] and Jorge Stolfi[3]

[*]Correspondence: hcgl@ic.uff.br
[1]Institute of Computing, Federal Fluminense University, Av. Gal. Milton Tavares de Souza, Sao Domingos, Niterói, Brazil
Full list of author information is available at the end of the article

Abstract

We describe robust methods for evaluating the similarity of down-sampled DNA or RNA sequences. We achieve robust down-sampling by encoding the nucleotide sequence into a packed 3-channel representation and using signal filtering techniques with Gaussian-like smoothing kernels. By using a multi-scale approach, we achieve scalability without significant loss of signal quality along down-sampled strings. We show that the comparison methods are robust under simple mutations such as isolated single-nucleotide substitutions, scattered insertions or deletions of short nucleotide sequences, reliably discriminating homologous and non-homologous pairs of sequences. Moreover, our filtering approach ensures that all levels of the multi-scale pyramid (except the original sequence) are practically free from aliasing artifacts and have the same degree of smoothing.

Keywords: bio-sequence analysis; signal analysis; sequence comparison; multi-scale

1 Introduction

We address a basic problem of bioinformatics: namely, evaluating the similarity of two genomic sequences (RNA or DNA). The goal such comparisons is to detect apparent *homologous* subsequences, derived from a common ancestor sequence by multiple replication steps, in which the sequences were modified by insertion, deletion and replacement of nucleotides

Our proposed approach starts by encoding the DNA or RNA sequence as a three-channel numeric signal, where each nucleotide is represented by a triplet of integers. We then view each channel as a sampled signal, and use standard signal filtering techniques to remove its high-frequency components. Each sample triplet in the filtered sequence can be visualized as a point along a smooth curve in three-dimensional space, whose shape depends on the local density of each nucleotide type.

The main contribution of this paper is the observation that, with proper filtering, the encoded sequence can be downsampled to provide a shorter representation of the original one, without introducing aliasing artifacts. Thanks to the filtering, the downsampled sequence is little affected by shifts in the downsampling phase, such as would be caused by isolated nucleotide insertions or deletions, or embedding of the same nucleotide sequence in different contexts.

The filtering and downsampling process can be iterated to produce a multi-scale description of the original DNA sequence. At each step, the number of samples is halved, reducing the processing costs by half or more; and twice as many nucleotides can be inserted or deleted at some point of the original sequence without significant change in the filtered and downsampled one.

At any level of this multi-scale representation, two encoded and filtered signals can be compared by an approximate string matching algorithm based on the dynamic programming paradigm [1]. The algorithm quantifies the likelihood of two given DNA strings being homologous. The multi-scale representation allows long segments of DNA to be compared at a small fraction of the cost of comparing the two sequences directly. The proper filtering before each downsampling step is essential for the functining of this algorithm.

Although we consider here only the uni-directional matching of DNA sequences, the multi-scale comparison technique could be easily applied to other kinds of biosequences; and bi-directional matching can be handled by running the algorithm a second time, after reversing and complementing one of the sequences.

2 Related work

Several approaches were proposed in order to speed up the pairing and matching of DNA or RNA sequences, relying in the translation of the character string representation to alternative forms of representation.

Ravichandran *et al.* [2] proposed a query-based alignment method for biological sequences mapping sequences to sequences to time-domain waveforms and then processing the waveforms for alignment in the time-frequency plane. This work was extended by Machado *et al.* [3] applying time-frequency analysis by wavelet decomposition to human DNA and protein sequences.

A graphical method based on dinucleotides and their positional information was proposed by Bari *et al.* [4], presenting a graphical representation of DNA sequences based on nucleotide ring structure. In the proposed representation, DNA sequences were converted into 16 dinucleotides on the surface of the hexagon.

The multi-scale analysis of DNA sequences was initially proposed by Futschik *et al.* [5] and Knijnenburg *et al.* [6], which employed a multi-scale segmentation of the sequences. Both works converted the analysed sequences into a single-channel numerical signal z extracted from the sequence. Knijnenburg *et al.* [6] defined z as the physical distance from each point of the sequence to a functional genomic element, and applied to it a multi-scale segmentation algorithm by Vicken *et al.* [7]. Futschik *et al.* [5] instead defined z as the $\mathtt{G}+\mathtt{C}$ content, and used multi-scale statistical analysis to obtain the segmentation.

3 Tetrahedral encoding of DNA

DNA and RNA sequences are commonly represented as a sequence of letters from the alphabet $\mathcal{B} = \{\mathtt{A},\mathtt{T},\mathtt{C},\mathtt{G}\}$ denoting the four nucleotides that may appear in DNA (with `U` instead `T` for RNA). Since our methods require arithmetical operations on sequence elements, like averaging and interpolation, we map each nucleotide to a point of three-dimensional space $\mathbb{R}^3$.

Like Anastassiou [8], we encode each DNA letter by a distinct vertex of a regular tetrahedron $\mathbb{T}^3$ in $\mathbb{R}^3$. However we position the tetrahedron so that all vertex coordinates are $+1$ or -1, namely

$$\begin{array}{lcl} \mathtt{A} & \rightarrow & (+1,+1,-1) \\ \mathtt{T} & \rightarrow & (+1,-1,+1) \\ \mathtt{C} & \rightarrow & (-1,+1,+1) \\ \mathtt{G} & \rightarrow & (-1,-1,-1) \end{array} \qquad (1)$$

See figure 1. A discussion of alternative encodings can be found in a earlier version of this work [9].

We will use the words *datum* for each element $x[j]$ of such an encoded sequence (a point of $\mathbb{R}^3$), and *sample* for each of its three coordinates. We assume that the index j runs from 0 to $n-1$, where n is the number of datums in the sequence. Note that a datum sequence can be viewed as a three separate sequences of numeric samples, that is, a three-channel discrete signal.

4 Filtering and down-sampling

4.1 Aliasing

By *down-sampling* a discrete signal x we mean assembling another signal x' by taking one every δ samples. The integer δ is the *downsampling stride*. Before doing that, we must make sure that the sequence x contains no Fourier components whose frequencies are at or above the Nyquist limit (one cycle every δ samples). Otherwise, the down-sampling will turn those high-frequency components of x into low-frequency components of x', which will be impossible to separate from the genuine low-frequency components of x'. (This phenomenon is known as *frequency aliasing* in signal theory.) Worse, the down-sampled sequence x' will vary drastically if the sequence x gets shifted by one position.

For instance, consider the two DNA sequences

$$\begin{aligned} X &= (\mathtt{A},\mathtt{T},\mathtt{A},\mathtt{G},\mathtt{T},\mathtt{C},\mathtt{G},\mathtt{C},\mathtt{C},\mathtt{A}) \\ Y &= (\mathtt{T},\mathtt{A},\mathtt{G},\mathtt{T},\mathtt{C},\mathtt{G},\mathtt{C},\mathtt{C},\mathtt{A},\mathtt{C}) \end{aligned} \tag{2}$$

Note that the sequence Y is basically X shifted 1 base to the left. If we down-sample both sequences by taking only the letters with even indices ($\delta = 2$), we would get $X' = (\mathtt{A},\mathtt{A},\mathtt{T},\mathtt{G},\mathtt{C})$ and $Y' = (\mathtt{T},\mathtt{G},\mathtt{C},\mathtt{C})$. Now Y' appears to be X' shifted 2 bases to the left, which would imply a shift of 4 bases at scale 0.

If the down-sampled sequence is obtained by averaging adjacent samples, namely if $x'[i] = (x[2i] + x[2i+1])/2$, the aliasing problem is somewhat reduced, but still present. For example, consider the two numeric sequences

$$\begin{aligned} x &= (0,2,2,0,0,2,2,0,0,2,2,0) \\ y &= (2,2,0,0,2,2,0,0,2,2,0,0) \end{aligned} \tag{3}$$

The sequences obtained by averaging pairs of consecutive samples and down-sampling with step 2 would be $x' = (1,1,1,1,1,1)$ and $y' = (2,0,2,0,2,0)$.

4.2 Convolution filtering

In order to avoid aliasing artifacts, we apply a smoothing convolution filter to a sequence x before down-sampling it. The filtering is defined by a *kernel radius* L and a table w of *kernel weights* $w[r]$ where $-L \le r \le +L$. Namely,

$$x'[i] = \sum_{r=-L}^{+L} w[r]\, x[\delta i + \sigma - r] \tag{4}$$

where σ is a suitable *downsampling offset*.

Formula 4 is to be applied for all indices j such that all indices in the right-and side are valid. Therefore, the offset σ must satisfy $L \leq \sigma < L+\delta$, and, if x has n samples, the length of x' will be $n' = \lfloor (n-\sigma-L+1)/\delta \rfloor$; unless $n \leq \sigma + L$, in which case x' is empty ($n' = 0$) by definition.

The kernel weights are usually positive, symmetric ($w[-r] = w[r]$) and decrease with increasing $|r|$. Therefore, each filtered sample $x'[i]$ is the weighted average of original samples $x[j]$ in a "fuzzy" window centered at sample $x[\delta i + \sigma]$, which has the largest weight in that average.

In particular, the first and last samples of the new sequence x' are the local averages of samples $x[\sigma]$ and $x[\delta(n'-1)+\sigma]$ of the original one. In other words, the filtering, as defined above, trims between L and $L+\delta-1$ samples from each end of the sequence x, before the downsampling proper (which reduces the length of the remaining sequence by a factor of about $1/\delta$).

The radius L must be at least $\lfloor \delta/2 \rfloor$, so that there is no gap between the windows of successive datums $x'[i]$ and $x'[i+1]$. Moreover, every sample $x[j]$ must give the same total contribution to x'; that is,

$$\sum_{k=-\infty}^{+\infty} w[j+k\delta] = 1/\delta \tag{5}$$

for any j in $\{0 \,..\, \delta-1\}$, with the convention that $w[r]$ is zero if $|r| > L$.

4.3 Multi-scale analysis of signals

In multi-scale signal analysis, a given discrete numerical sequence x is transformed into a *hierarchy* or *pyramid* of discrete signals $x^{(0)}, .. \, x^{(h)}$; where $x^{(0)}$ is the original signal x, and each subsequent signal $x^{(k)}$ with $k \geq 1$ is a down-sampled and filtered version of the previous one $x^{(k-1)}$. In principle one can use different parameters $L^{(k)}, \delta^{(k)}, \sigma^{(k)}$, and a different table of kernel weights $w^{(k)}$ for eack level. The lengths of these sequences will be $n^{(k)}$, where $n^{(0)} = n$ is the length of x, and $n^{(k)} = \lfloor (n^{(k-1)} - \sigma^{(k)} - L^{(k)} + 1)/\delta^{(k)} \rfloor$ for $k \geq 1$

Each level $x^{(k)}$ of the pyramid can also be seen as the result of filtering and downsampling the original sequence $x = x^{(0)}$ with the *cumulative parameters* $\delta^{*(k)}, \sigma^{*(k)}, L^{*(k)}$ and a *cumulative kernel weight table* $w^{*(k)}$. For the first level we have $\delta^{*(1)} = \delta^{(1)}$, $\sigma^{*(1)} = \sigma^{(1)}$, $L^{*(1)} = L^{(1)}$, and $w^{*(1)} = w^{(1)}$. For $k \geq 2$, the cumulative parameters are defined recursively as

$$\begin{aligned} \delta^{*(k)} &= \delta^{(k)} \delta^{*(k-1)} \\ \sigma^{*(k)} &= \sigma^{(k)} \delta^{*(k-1)} + \sigma^{*(k-1)} \\ L^{*(k)} &= L^{(k)} \delta^{*(k-1)} + L^{*(k-1)} \end{aligned} \tag{6}$$

The cumulative weights $w^{*(k)}$ are similarly defined as the convolution of $w^{*(k-1)}$ with $w^{(k)}$, after the latter has been "stretched" by inserting $\delta^{*(k-1)} - 1$ zeros between succesive weights.

4.4 Multi-scale analysis of DNA sequences

With the encoding described in section 3, multi-scale analysis can be applied to DNA sequences as well. Specifically, each original DNA sequence X is transformed by the encoding into a datum sequence $x = x^{(0)}$ with the same length, and from these we derive

$x^{(1)}, .. x^{(h)}$, by filtering each channel as a numeric discrete signal, and downsampling the datum sequence. See figure 5. We found it convenient to use $\delta^{(1)} = 1$ and $\sigma^{(1)} = L^{(1)}$, so that level $z^{(1)}$ is merely a smoothed and truncated version of $x^{(0)}$, without downsampling; and $\delta^{(k)} = 2$ for all $k \geq 2$.

The kernel weights that we use are $w^{(k)}[r] = W^{(k)}[r]/D^{(k)}$, where $W^{(k)}$, $D^{(k)}$, and the radii $L^{(k)}$are given in table 1. The power spectra of these kernels are shown in figure 3.

We deine the *degree of smoothing* $U^{(k)}$ of each level k by the recurrence $U^{(0)} = 0$ and $U^{(k)} = (U^{(k-1)} + V^{(k)})/(\delta^{(k)})^2$ for all $k \geq 1$; where $V^{(k)}$ is the variance of the filtering kernel $w^{(k)}$, interpreted as a probability distribution on the indices $\{-L^{(k)} .. +L^{(k)}\}$. See figure 4.

The quantity $U^{(k)}$ is an estimate of the variance of the impulse response function of the linear process that transforms the unfiltered sequence $x^{(0)}$ into $x^{(k)}$. This process is not shift-invariant, but, with the filtering kernels of table 1, the response for each input sample is very close to a Gaussian hump with variance $U^{(k)}$ and a fractional mean. In particular, the definition $U^{(0)} = 0$ is consistent with the fact that the original unfiltered sequence $x^{(0)}$ is not smooth at all. With our choices of kernels and steps, this recurrence gives $U^{(k)} = 2.00$ for all $k \geq 1$. We take this to mean that all scales are smoothed to the same degree, and equally safe from aliasing artefacts.

4.5 Visualizing DNA sequences as space curves

Each level $x^{(k)}$ of the multi-scale hierarchy of a DNA sequence is a sequence of $n^{(k)}$ points in three-dimensional space. These points can be interpolated with a cubic spline for any real argument t in the range $[0 \mapsto n^{(k)} - 1]$, to yield a smooth curve $x^{(k)}(t)$ in three-dimensional space.

Each level $x^{(k)}$ of the multi-scale hierarchy of a DNA sequence is a sequence of points $x^{(k)}[0], .. x^{(k)}[n-1]$ in three-dimensional space. These points can be interpolated with a cubic spline for any real argument t in the range $[0 \mapsto n-1]$, to yield a smooth curve $x^{(k)}(t)$ in three-dimensional space. This curve can be plotted with arbitrary 3D rendering methods or viewed with interactive 3D visualization tools. See figure 2.

For $k = 0$, the curve intersects itself at a tetrahedron vertex at every integer t, and therefore is quite uninformative; but for $k \geq 1$ self-intersections are rare, and the general shape of the curve conveys useful information, as we shall see. At successive stages, the curve becomes necessarily simpler, losing the smaller details (and being trimmed at each end) while retaining the larger ones. See figure 6. These curves can be effective tools for visual comparison of sequences with up to a couple hundred datums [9].

5 Comparison of filtered DNA sequences

We now proceed to describe the comparison of DNA sequences that have been encoded, filtered, and down-sampled as described in the precious sections.

5.1 Evolution model

labels.evomodel

Recall that the goal of biosequence comparison is to detect homologous subsequences, that are derived from the same ancestral sequence. Specifically, we assume that the two sequences X and Y to be compared were independently created from some ancestral sequence Z by multiple biological replications, and suffered several evolutionary *events* during this process; where each event may be

- a *point mutation* that replaces a single nucleotide by a different one;
- a *short deletion* that deleted a few consecutive nucleotides; or
- a *short insertion* that inserts a few nucleotides between two consecutive ones.

In real genomes one may also have *rearrangement* events, in which the DNA chain is broken into several relatively large pieces that are reattached in a different order, possibly with some pieces being lost, duplicated, or imported from some "foreign" DNA. Our algorithm does not try to model, identify, or account for such rearrangements. If X and Y contain some ancestral substring Z that was modified by such events, our algorithm will hopefully report it as several homologous pairs of sub-strings (X'_1, Y'_1), (X'_2, Y'_2), ..., where each pair (X'_j, Y'_j) is derived from a maximal segment of Z that was not split by any of the rearrangement events.

Any string can be turned into any other string by a series of single-letter insertions and deletions; but the inclusion of the three classes of events above can be justified by biology and statistics. Namely, we assume that

- the number of evolutionary events that occurred since the common ancestor is small compared to the length of the two strings;
- events occur independently and at random places in the string, with uniform probability;
- the probability of a point mutation is substantially greater than that of a deletion and insertion in the same spot; and
- the probability of a short insertion or deletion decays exponentially with the number of nucleotides added or lost.

These assumptions imply that the letters of two homologous strings X, Y can be paired, preserving their order, so that paired letters are equal; except for a relatively few places where the paiting is disturbed by short insertions, and short deletions. In contrast, non-homologous strings are as dissimilar as any two random sub-strings of a genome can be. Like most homology detection methods, our algorithm is based on this hypothesis: that two maximal sub-strings that are sufficiently similar, in this sense, are likely to be homologous.

Our comparison criterion is meant to operate on versions x, y of the original strings that have been numerically encoded, filtered, and down-sampled, as described in section 4.4. Even so, the two sequences can be compared by a variant of the well-known dynamic programming algorithm that finds the longest common subsequnce of two strings. The running time of that algorithm is proportional to the product of the lengths of the two strings. Therefore, if x and y contain only one sample for every K letters of X and Y, the comparison will require about $1/K^2$ as much memory and $1/K^2$ or $1/K^3$ as much computing time, relative to the cost of comparing X and Y directly.

6 Pairings

Formally, we define a *pairing* between two arbitrary sequences x and y as a sequence of pairs $(r_0, s_0), .. (r_p, s_p)$, where each r_i is an index into x, and each s_i is an index into y.

If x and y are raw nucleotide sequences X and Y, each pair (r_i, s_i) can be interpreted as a hypothesis that the nucleotides $X[r_i]$ and $Y[s_i]$ are homologous, that is, are replicas of the same nucleotide of the hypothetical common ancestral sequence. If x and y are numerically encoded, filtered, and down-sampled versions of X and Y, then each pair (r_i, s_i) can be interpreted as saying that the part of X summarized by datum $x[r_i]$ and the part of Y summarized by datum $y[s_i]$ probably contain a significant number of homologous letter pairs.

Each pair (r_i, s_i) is called a *rung* of the pairing. The sum $r_i + s_i$ is the *position* of the rung, and the difference $s_i - r_i$ is its *offset*.

Each pair of successive rungs (r_i, s_i) and (r_{i+1}, s_{i+1}) is said to be a *step* of the pairing. Since we are not considering segment rearrangements, we require the pairing to be *strictly monotonic*, meaning that it must satisfy $r_i < r_{i+1}$ and $s_i < s_{i+1}$ in every step. Note that the pairing has p steps and $p+1$ rungs.

The pairing is said to *span* the substrings of x and y that starts with elements $x[r_0]$ and $y[s_0]$ and end with elements $x[r_p]$ and $y[s_p]$, respectively.

6.1 Perfect and connected pairings

A *perfect pairing* is one where every step is *normal*, that is, $r_{i+1} = r_i + 1$ and $s_{i+1} = s_i + 1$. A step that is not normal is a *skipping step*, that leaves a *break* (one or more unpaired elements) in one or both of the two sequences, before the next matched pair of elements. Such steps are meant to model evolutionary events where a small stretch of DNA was inserted into one sequence or deleted from the other. (We do not attempt to distinguish between these two possibilities.)

In this work we consider only pairings that are *connected*, namely where every step $(r_i, s_i) \to (r_{i+1}, s_{i+1})$ satisfies $r_{i+1} = r_i + 1$ or $s_{i+1} = s_i + 1$. In other words, at each step the pairing may have a break on either sequence, but not on both at the same time. A step that skips bases on both sequences would represent either a multi-base substitution (deletion of a string of bases and insertion of an unrelated string) in one of the sequences, or insertion of two unrelated strings on both sequences at about the same spot. Both events are assumed to be too rare to consider in the initial search for homologous sub-strings.

The problem we are solving can be now restated as follows: given two biosequences X and Y, respectively with m and n letters, find a connected strictly monotonic pairing r, s between them, consisting mostly of normal steps, such that the corresponding subsequences X', Y' (defined by $X'[i] = X[r_i]$ and $Y'[i] = Y[s_i]$) are sufficiently similar. In the process of solving this problem for the original biosequences X and Y, we solve the same problem at each scale k, for the filtered and downsampled sequences $x^{(k)}$ and $y^{(k)}$.

6.2 Likelihood of a DNA sequence pairing

Our algorithm implicitly requires the assignment of numerical quality scores to a given candidate pairing r, s between two filtered and downsampled sequences x, y. In this evaluation, we must consider the similarity of the paired datums $x[r_i]$ and $y[s_i]$, and the number of imperfections (skipping steps and unpaired bases) in the pairing itself. In this and the following functions, we derive our scoring function based on the *log likelihood* criterion which underlies the scoring functions for DNA alignment often used in computational biology.

Let's consider first the evaluation of a pairing r, s, with p steps, between two unfiltered DNA sequences X, Y. Let $\Pr_{\mathsf{H}}((X, r) \leftrightarrow (Y, s))$ denote the likelihood that the substrings of X and Y spanned by the pairing are homologous—that is, descend from a common ancestral DNA sequence Z. We use the simplistic formula

$$\Pr_{\mathsf{H}}((X, r) \leftrightarrow (Y, s)) = \left(\prod_{i=0}^{p} \Pr_{\mathsf{P}}(X[r_i] \leftrightarrow Y[s_i]) \right) \left(\prod_{i=1}^{p} \Pr_{\mathsf{I}}(r_i - r_{i-1}, s_i - s_{i-1}) \right) \quad (7)$$

Here, the *nucleotide pairing factor* $\mathrm{Pr_P}(X[r_i] \leftrightarrow Y[s_i])$, associated to each rung (r_i, s_i) of the pairing, can be interpreted as the probability that the same ancestral nucleotide $Z[k]$ yielded $X[r_i]$ in one sequence and $Y[s_i]$ in the other, accounting for possible point mutations. We assume that this factor depends only on the two nucleotides (and therefore could be provided by a table with 4×4 entries). In particular, if we assume that any nucleotide may mutate into any other with the same probability μ, then

$$\mathrm{Pr_P}(X[r_i], Y[s_i]) = \begin{cases} (1-\mu)^2 + \frac{1}{3}\mu^2 & \text{if } X[r_i] = Y[s_i], \text{ and} \\ 2\mu(1-mu) + \frac{2}{3}\mu^2 & \text{if } X[r_i] \neq Y[s_i]. \end{cases} \tag{8}$$

For example, if $\mu = 0.1$, then $\mathrm{Pr_P}(X[r_i] \leftrightarrow Y[s_i])$ is ≈ 0.8133 if the nucleotides are the same, and ≈ 0.1867 if they are different.

The *insertion/deletion factor* $\mathrm{Pr_I}(r_i - r_{i-1}, s_i - s_{i-1})$ in formula (7), associated with a step $(r_{i-1}, s_{i-1}) \to (r_i, s_i)$, can be interpreted as the probability of the step advancing $r_i - r_{i-1}$ nucleotide positions in sequence X and $s_i - s_{i-1}$ nucleotides in sequence Y. Recall that we are considering only connected pairings, so at least one of these differences is 1; so can write the step-associated factor as $\mathrm{Pr_I}(\lambda_i)$ where $\lambda_i = \max\{r_i - r_{i-1} - 1, s_i - s_{i-1} - 1\}$ is the length of the presumed insertion or deletion, i. e. the number of nucleotides that are skipped and left unpaired in that step. We assume that $\mathrm{Pr_I}(\lambda_i)$ decreases exponentially when $\lambda_i \geq 1$. Namely,

$$\mathrm{Pr_I}(\lambda_i) = \begin{cases} 1 - \eta & \text{if } \lambda_i = 0 \text{ (the step is normal), and} \\ \eta \tau^{\lambda_i - 1} \frac{1-\tau}{1-\tau^{\lambda_{\max}}} & \text{if } 1 \leq \lambda_i \leq \lambda_{\max}. \end{cases} \tag{9}$$

where η if the probability of an insertion or deletion occurring at all after any nucleotide, $\lambda_{\max}$ is the maximum number of nucleotide insertions or deletions considered at one spot, and τ is the probability factor for each additional unpaired nucleotide. For example, if $\eta = 0.05$, $\tau = 0.90$, and $\lambda_{\max} = 20$, then $\mathrm{Pr_I}(1) = 0.95$, $\mathrm{Pr_I}(2) \approx 0.00578$m $\mathrm{Pr_I}(3) \approx 0.00520$, $\mathrm{Pr_I}(4) \approx 0.00468, \ldots, \mathrm{Pr_I}(20) \approx 0.00087$.

6.3 Similarity score for paired DNA sequences

We now define the *similarity* of two unfiltered nucleotide sequences X and Y under the pairing r, s as the negative of the logarithm of formula (7), namely

$$S_{\mathrm{H}}(X, r, Y, s) = \sum_{i=0}^{p} S_{\mathrm{P}}(X[r_i], Y[s_i]) + \sum_{i=1}^{p} S_{\mathrm{I}}(\lambda_i) \tag{10}$$

where

$$S_{\mathrm{P}}(X[r_i], Y[s_i]) = \begin{cases} w_{\mathrm{E}} & \text{if } X[r_i] = Y[s_i], \text{ and} \\ w_{\mathrm{D}} & \text{if } X[r_i] \neq Y[s_i] \end{cases} \tag{11}$$

with $w_{\mathrm{E}} = -\log((1-\mu)^2 + \frac{1}{3}\mu^2)$, $w_{\mathrm{D}} = -\log(\mu(1-mu) + \frac{2}{3}\mu^2)$; and

$$S_{\mathrm{I}}(\lambda_i) = \begin{cases} w_{\mathrm{N}} & \text{if } \lambda_i = 0, \text{ and} \\ w_{\mathrm{B}} + (\lambda_i - 1) w_{\mathrm{S}} & \text{if } 1 \leq \lambda_i \leq \lambda_{\max} \end{cases} \tag{12}$$

with $w_{\mathsf{N}} = -\log(1-\eta)$, $w_{\mathsf{B}} = -log(\eta(1-\tau)/(1-\tau^{\lambda_{\max}}))$, and $w_{\mathsf{S}} = -log(\tau)$.

Formula (10) can be written also as

$$S_{\mathsf{H}}(X,r,Y,s) = w_{\mathsf{E}} n_{\mathsf{E}}(X,r,Y,s) + w_{\mathsf{D}} n_{\mathsf{D}}(X,r,Y,s) + w_{\mathsf{N}} n_{\mathsf{N}}(r,s) + w_{\mathsf{B}} n_{\mathsf{B}}(r,s) + w_{\mathsf{S}} n_{\mathsf{S}}(r,s) \quad (13)$$

where w_{E}, w_{D}, w_{N}, w_{B}, and w_{S} are the real coefficients defined above, and

- $n_{\mathsf{E}}(X,r,Y,s)$ is the number of equal nucleotides paired by r,s, that is, $\sum_{i=0}^{p}(X[r_i] = Y[s_i])$;
- $n_{\mathsf{D}}(X,r,Y,s)$ is the number of unequal paired nucleotides, that is, that is, $\sum_{i=0}^{p}(X[r_i] \neq Y[s_i])$;
- $n_{\mathsf{N}}(r,s)$ is the number of normal steps of the pairing, namely $\sum_{i=1}^{p}((r_i - r_{i-1} = 1) \wedge (s_i - s_{i-1} = 1))$;
- $n_{\mathsf{B}}(r,s)$ is the number of breaks (skipping steps), namely $\sum_{i=1}^{p}((r_i - r_{i-1} > 1) \vee (s_i - s_{i-1} > 1))$;
- $n_{\mathsf{S}}(r,s)$ is the total number of nucleotides that were left unpaired in both sequences, namely $\sum_{i=1}^{p}((r_i - r_{i-1} - 1) + (s_i - s_{i-1} - 1))$;

Note that $n_{\mathsf{D}}(X,r,Y,s) = p - n_{\mathsf{E}}(X,r,Y,s)$, and $n_{\mathsf{N}}(r,s) = p - 1 - n_{\mathsf{B}}(r,s)$.

Since the logarithm is a monotone function, the similarity $S_{\mathsf{H}}(X,r,Y,s)$ defined by formula (13) is minimum when the likelihood of the pairing defined by formula (7) is maximum.

6.4 Optimum pairings for DNA sequences

It is well-known that the optimum (maximum-score) pairing between two DNA sequnces X,Y, with any additive local scoring criterion, can be found with the dynamic programming algorithm [1].

However, if the coefficients of the scoring formulas (11)– (6.5) and (13) (w_{E}, w_{D}, w_{B}, w_{S}, and w_{N}) are defined as in section 6.2, they will be all positive. Then the maximum-score pairing between two sequences x,y, respectively with m and n datums, will then always have maximal length: that is, $r_0 = 0$ or $s_0 = 0$, and $r_p = m-1$ or $s_p = n-1$. That is because the score increases even when one extends the pairing with a rung between two dissimilar elements, or a skipping step.

We avoid this inconvenient behavior by the standard trick of subtracting from the score a bias term proportional to the total number $\lambda = (r_p - r_0 + 1) + (s_p - s_0 + 1)$ of datums spanned by the pairing on both sequences. We also subtract a constant bias w_{C}, to ensure that a pairing will not have a positive score unless it has a certain minimum span λ.

Since $n_{\mathsf{E}} + n_{\mathsf{D}}$ is the number of rungs p, it can be seen that $\lambda = 2(n_{\mathsf{E}} + n_{\mathsf{D}}) + n_{\mathsf{S}}$. Therefore, the effect we seek can be implemented by formulas (11)– (6.5) and (13), after subtracting suitable bias from the coefficients w_{E}, w_{D}, and w_{S}, and subtracting the constant bias term w_{C}.

The coefficient w_{D} and w_{S} must be negative, but w_{E} positive; so that the total score decreases when the pairing is extended with a discrepant rung or a skipping step, but increases when it is extended with a normal step and a rung between equal nucleotides.

The coefficient w_{B} should be negative too in order to penalize breaks in the sequences. The coefficient w_{B} should be larger than w_{D} in absolute value, reflecting the assumption that insertions and deletions are less common than single-nucleotide replacements. On the other hand, w_{S} should be much smaller than w_{D} in absolute value, reflecting the assumption that the probability of insertions/deletions varies little with their length.

With the proper coefficient values, the score for the optimum pairing between homologous sequences should be positive and should increase as their total length increases; while even the best pairing between unrelated sequences should be negative. Good values for the weights can be determined iteratively by computing the optimum pairing for homologous and non-homologous sequences, and adjusting the weights to as to get the best separation between the two classes.

6.5 Similarity score for paired datum sequences

let x and y be filtered sequences of numeric datums. A pairing r,s between them represents an approximate coarse-scale pairing between the underlying DNA strings X and Y. Namely, suppose that successive datums in x (or y) correspond to places in X (or in Y) that are δ nucleotides apart. A rung (r_i, s_i) of the pairing can be interpreted as the tentative hypothesis that the segment of X summarized by datum $x[r_i]$ and the segment of Y summarized by $y[s_i]$ homologous, apart from a relative displacement of about $\pm\delta/2$.

The likelihood of that hypothesis can be roughly estimated from the datums $x[r_i]$ and $y[s_i]$, because they describe the local density of each nucleotide, at certain places in each sequence. Although similar nucleotide densities do not imply homology, *dis*similar densities make homology less likely.

Specifically, suppose that the datum d is an element $x[i]$ of a sequence x that was obtained by encoding, filtering, and downsampling a DNA sequence X. Let $X[k]$ be an element in string X at the center of the window that is summarized by datum $x[i]$. Let U be any of the four nucleotides $\mathtt{A},\mathtt{T},\mathtt{C},\mathtt{G}$, and let $u = (u_0, u_1, u_2)$ be its numeric code per table (1). We can estimate the probability that $X[k]$ is U by the formula

$$\Pr(X[k] = U|d) = \frac{1 + d_0u_0 + d_1u_1 + d_2u_2}{4} \tag{14}$$

In particular, if d is one of the corners of the tetrahedron, formula (14) is categorical on the corresponding nucleotide. E. g., if $d = (+1,+1,-1)$, then $\Pr(X[k] = \mathtt{A}|d) = 1$ and $\Pr(X[k] = \mathtt{T}|d) = \Pr(X[k] = \mathtt{C}|d) = \Pr(X[k] = \mathtt{G}|d) = 0$. Moreover, if d is $(+1,0,0)$, at the midpoint of the edge between the `A` and `T` corners, then $\Pr(X[k] = \mathtt{A}|d) = \Pr(X[k] = \mathtt{T}|d) = 1/2$ and $\Pr(X[k] = \mathtt{C}|d) = \Pr(X[k] = \mathtt{G}|d) = 0$. If d is at the center of a face, e. g. $d = (+1/3,+1/3,+1/3)$, the the three corner nucleotides get probability $1/3$, and the fourth one gets probability zero. Finally, if d is the center $(0,0,0)$ of the tetrahedron, formula (14) assigns the same probability $1/4$ to each nucleotide.

From that, we can estimate the probability that the two original nucleotides are the same, that is

$$\Pr(X[k] = Y[l]|d,e) = \sum_{U\in\mathcal{B}} \Pr(X[k] = U|d)\Pr(Y[l] = U|e) \tag{15}$$

Using formula (14), formula (15) evaluates to

$$\Pr(X[k] = Y[l]|d,e) = \frac{1 + d_0e_0 + d_1e_1 + d_2e_2}{4} = \frac{1 + d\cdot e}{4} \tag{16}$$

Note that the maximum value of $d\cdot e$ is $+3$, which occurs (only) when d and e are the same corner; in that case (only), formula (16) gives $Pr(X[k] = Y[l]|d,e) = 1$. The minimum value

of the dot product $d \cdot e$ for d, e in the tetrahedron $\mathbb{T}^3$ is -1, which occurs when d is a corner and e is anywhere on the opposite face, or when d lies on some edge and e lies anywhere on the opposite edge. In that case (only), formula (16) gives $Pr(X[k] = Y[l]|d,e) = 0$.

For any other combination of d and e, formula (16) gives a probability $Pr(X[k] = Y[l]|d,e)$ strictly between 0 and 1. In particular, if both samples are at the center of T^3 — that is, $d = e = (0,0,0)$ — then $Pr(X[k] = Y[l]|d,e) = 1/4$; as one woudl expect if the nucleotides $X[k]$ and $Y[l]$ were randomly and independently drawn with equal probabilities.

We now can generalize the scoring function S_{H} of formula (10) to pairings between datum sequences instead of DNA sequences. For that purpose, we use the same scoring formulas (10) and (13), namely

$$S_{\mathsf{H}}(x,r,y,s) = \sum_{i=0}^{p} S_{\mathsf{P}}(x[r_i], y[s_i]) + \sum_{i=1}^{p} S_{\mathsf{I}}(\lambda_i) \tag{17}$$

The insertion/deletion term S_{I}, that depends only on the pairing, is the same as before, given by formula . The pairing score term S_{P}, on the other hand, must be generalized for datum triplets with fractional samples, rather than nucleotides. We define $S_{\mathsf{P}}(d,e)$ as the expected value of $S_{\mathsf{P}}(X[k], Y[l])$ as defined by formula (11), namely

$$S_{\mathsf{P}}(d,e) = w_{\mathsf{E}} \Pr(X[k] = Y[l]|d,e) + w_{\mathsf{D}} (1 - \Pr(X[k] = Y[l]|d,e)) = w_{\mathsf{E}} \frac{1 + d \cdot e}{4} + w_{\mathsf{D}} \frac{3 - d \cdot e}{4} \tag{18}$$

Then formula (13) also holds, except that n_{E} and n_{D} must be replaced by

$$n_{\mathsf{E}}(X,r,Y,s) = \sum_{i=0}^{p} \Pr(X[k] = Y[l]|x[r_i], y[s_i]) \tag{19}$$

$$n_{\mathsf{D}}(X,r,Y,s) = \sum_{i=0}^{p} (1 - \Pr(X[k] \neq Y[l]|x[r_i], y[s_i])) \tag{20}$$

with $\Pr(X[k] = Y[l]|x[r_i], y[s_i])$ given by formula (16).

With proper coefficients, the score $S_{\mathsf{H}}(x,r,y,s)$ easily differentiates correct pairings of homologous sequences from pairings between non-homologous sequences, provided the pairing is long enough. See figure 7.

The values of the coefficients $(w_{\mathsf{E}}, w_{\mathsf{D}}, w_{\mathsf{N}}, w_{\mathsf{B}}, w_{\mathsf{S}}, w_{\mathsf{C}})$ must be different for each scale. Note that as the filtering level increases the sample values are averages taken over increasingly wider intervals. Moreover, at scale k any break that skips less than 2^k samples on each sequence will disappear.

7 Multi-scale matching of DNA sequences

We now describe briefly the use of multi-scale analyisis to efficiently identify homologous sub-sequences in to DNA or RNA sequences X and Y. The method is similar to the multi-scale algorithm that we developed some time ago to efficiently find matching segments on the outlines of pottery fragments [10, 11]. For lack of space, the details will have to be described in a separate paper.

The dynamic programming (DP) agorithm [1] can be used to find the optimum pairing (r,s) of two filtered and downsampled datum sequences x and y derived from X and Y. That

pairing can be mapped into a pairing (r', s') between the original strings, which can be used as a rough guess for computing the optimum pairing between them, with substantial savings of processing time.

Specifically, recall that the original DP algorithm builds a matrix M with one element $M[i, j]$ for each index i of X and each index j of Y. Its cost is proportional to $nm\lambda_{\max}$, where n and m are the lengths of the two original strings, with $m \leq n$, and $\lambda_{\max}$ is the maximum length of an insertion to be considered. We instead use a modification of the DP algorithm that computes only a relatively small subset of the elements of the matrix M. These elements comprise a narrow strip that surrounds the elements $M[r'_k, s'_k]$ that are specified by the rungs of the pairing (r', s'). The cost of this *pairing refinement algorithm* is only $n\lambda^2_{\max}$, or $\lambda_{\max}/m$ times the cost of the full DP algorithm.

This technique can be applied at each level of a multi-scale hierarchy of filtered and down-sampled signals, from coarsest to finest. That is, we run the full DP algorithm at some sufficiently coarse scale k, where the original sequences have been downsampled with step $\delta^{*(k)}$, and the maximum insertion length has been divided by that factor too. The DP processing cost will then be reduced by a factor $1/(\delta^{*(k)})^3$, relative to what it would cost if applied to the original sequences. Then we map the optimum pairing found to the next finer scale $k-1$, and we refine it, as described above, to obtain an optimum pairing at that scale. This process is then repeated to scales $k-2, k-3, \ldots$, until we get a pairing for the level 0 (original) sequences.

As the sequences get filtered and downsampled at increasing scales, the mutations and insertions get averaged with the adjacent samples. Therefore, the similarity score for filtered versions $x^{(k)}$ and $y^{(k)}$ of two homologous DNA sequences may be relatively low, even under the optimum pairing; possibly lower than the score for two totally unrelated sequences. Therefore, it is usually necessary to keep multiple *candidate pairings* at each level, rather than just the onle with maximum score. When going from level k to level $k-1$, the algorithm maps and refines each candidate, then discards the half of the set with the lowest scores. As discussed in the pottery fragments paper [10], even with multiple candidates the processing time is still much less than that of the original DP algorithm applied at scale 0 directly.

8 Conclusions

We described formulas for comparing nucleotide sequences that have been numerically encoded, filtered, and sub-sampled. Tests indicate that the formulas can discriminate between homologous and non-homologous (random) pairs of sequences, and that the discrimination increases with the length of the sequences. These formulas can be used with the standard dynamic programming algorithm to find approximate optimal pairings between two presumed homologous sequences, at a fraction of the cost of running the algorithm on the original sequences.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

Author details

[1] Institute of Computing, Federal Fluminense University, Av. Gal. Milton Tavares de Souza, Sao Domingos, Niterói, Brazil.
[2] Technological Instute of Castilla y León, Pol. Industrial Industrial Villalonquejar, c/Lopez Bravo 70, 09001 Burgos, Spain.
[3] Institute of Computing, State University of Campinas, Av. Albert Einstein, 1251, Cidade Universitaria, Campinas, Brazil.

References

1. Schoniger, M., Waterman, W.M.: A local algorithm for DNA sequence alignment with inversions. Bulletin of Mathemathical Biology **54**(4), 521–536 (1992)
2. Ravichandran, L., Papandreou-Suppappola, A., Spanias, A., Lacroix, Z., Legendre, C.: Waveform mapping and time-frequency processing of DNA and protein sequences. IEEE Transactions on Signal Processing **59**(9), 4210–4224 (2011)
3. Machado, J.A.T., Costa, A.C., Quelhas, M.D.: Wavelet analysis of human DNA. Genomics **98**(3), 155–163 (2011)
4. Bari, A.G., Reaz, M.R., Islam, A.T., Choi, H.-J., Jeong, B.-S.: effective encoding for dna sequence visualization based on nucleotide's ring structure. Evolutionary bioinformatics online **9**, 251 (2013)
5. Futschik, A., Hotz, T., Munk, A., Sieling, H.: Multiscale DNA partitioning: Statistical evidence for segments. Bioinformatics, 180 (2014)
6. Knijnenburg, T.A., Ramsey, S.A., Berman, B.P., Kennedy, K.A., Smit, A.F.A., Wessels, L.F.A., Laird, P.W., Aderem, A., Shmulevich, I.: Multiscale representation of genomic signals. Nature Methods (2014)
7. Vincken, K.L., Koster, A.S.E., Viergever, M.A.: Probabilistic multiscale image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **19**(2), 109–120 (1997)
8. Anastassiou, D.: Digital signal processing of biomolecular sequences. Technical Report CU/EE/TR2000-20-042, Department of Electrical Engineering, Columbia University (2002)
9. Leitão, H.C.G., Saracchini, R.F.V., Stolfi, J.: Geometric encoding, filtering, and visualization of genomic sequences. In: Proceedings of the 6th International Conference on Information Visualization Theory and Applications, pp. 219–224 (2015)
10. Leitão, H.C.G., Stolfi, J.: A multiscale method for the reassembly of two-dimensional fragmented objects. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(9), 1239–1251 (2002). doi:10.1109/TPAMI.2002.1033215
11. da Gama Leitão, H.C.: Reconstrução automática de objetos fragmentados. PhD thesis, Institute of Computing, University of Campinas, Campinas, SP, Brazil (November 1999). (In Portuguese)
12. Harris, R.S.: Improved pairwise alignment of genomic DNA. PhD thesis, The Pennsylvania State University (2007)

Figures

Missing figure misc/tetra/main.eps

Figure 1: The tetrahedron $\mathbb{T}^3$ whose corners encode the letters of the DNA alphabet $\mathcal{B}$.

Missing figure tfil/250E-wg020-wg060-02.eps

Figure 2: Three-dimensional plot of a DNA segment from a *Drosophila sp.* genome, originally with 250 nucleotides, filtered by the $w^{(1)}$ filter of table 1, with no down-sampling, and then with the $w^{(2)}$ filter, down-sampled with step $\delta^{(2)} = 2$. The beads along the curve are the actual datums; the connecting lines were reconstructed by cubic interpolation. The entire curve was magnified by the scale factor $s = 1.440$ relative to the origin (the center of the tetrahedron) for clarity.

Missing figure spec/250A-wg020-wg060-00-f0.eps
Missing figure spec/250A-wg020-wg060-00-f1.eps

Figure 3: Power spectra of the filtering kernels $w^{(k)}$ of table 1, for the initial step $k = 1$ (top) and subsequent steps $k \geq 2$ (bottom).

Missing figure spec/250A-wg020-wg060-00-fo.eps
Missing figure spec/250A-wg020-wg060-01-fo.eps
Missing figure spec/250A-wg020-wg060-02-fo.eps
Missing figure spec/250A-wg020-wg060-03-fo.eps

Figure 4: Idealized power spectrum of an unfiltered periodic random binary signal with a 256-sample period (top) and its spectra after 1, 2, and 3 filtering steps. The vertical line shows the maximum frequency that is preserved without aliasing by the combined down-samplings from level 0 to the indicated level.

$x^{(0)}$ Missing figure filt/250A-wg020-wg060-00.eps
$x^{(1)}$ Missing figure filt/250A-wg020-wg060-01.eps
$x^{(2)}$ Missing figure filt/250A-wg020-wg060-02.eps
$x^{(3)}$ Missing figure filt/250A-wg020-wg060-03.eps

Figure 5: Multi-scale versions of a DNA sequence with 250 nucleotides, encoded as corners of $\mathbb{T}^3$, filtered and down-sampled as described in section 4.2 and 4.4. The three channels are plotted in red, green, and blue, respectively

Missing figure tfil/500A-wg020-wg060-01.eps $x^{(1)}$, $s = 1.000$
Missing figure tfil/500A-wg020-wg060-02.eps $x^{(2)}$, $s = 1.200$
Missing figure tfil/500A-wg020-wg060-03.eps $x^{(3)}$, $s = 1.440$
Missing figure tfil/500A-wg020-wg060-04.eps $x^{(4)}$, $s = 1.728$

Figure 6: Three-dimensional plots of a DNA segment from a *Drosophila sp.* genome, originally with 500 nucleotides, filtered and down-sampled at various scales by the filter kernels of table 1. Each curve was magnified by the indicated scale factor s, for clarity.

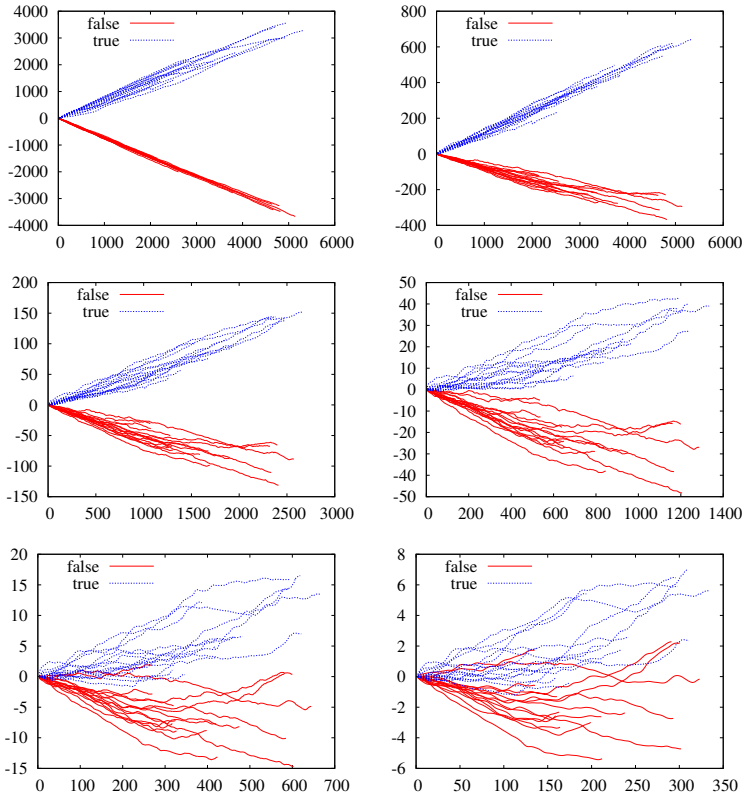

Figure 7: Similarity scores of increasingly longer segments of pairings between 15 pairs of homologous DNA sequences (blue) and 15 pairs of non-homologous sequences (red), as a function of length, at six scales of filtering (0–5). Each line corresponds to one pairing in the input data. Each point along the line corresponds to a centered segment of that pairing, with specified span shown on the horizontal axis. The original homologous sequences are fragments of the DNA of two *Drosophila* species, with at least 2048 bases and 85% or more equal nucleotides under the optimum pairing, identified and paired with the LastZ tool [12]. The non-homologous pairs are obtained by pairing those same DNA sequences at random.

Tables

Table 1: Elements of the filtering kernels used in the example. The last line $V^{(k)}$ is the variance of the kernel $w^{(k)}$, viewed as a probability distribution on the indices with mean 0.

	$k = 1$	$k \geq 2$
$L^{(k)}$	6	10
$D^{(k)}$	35440	61364
$W^{(k)}[0]$	9992	9992
$W^{(k)}[\pm 1]$	7786	9193
$W^{(k)}[\pm 2]$	3680	7161
$W^{(k)}[\pm 3]$	1055	4722
$W^{(k)}[\pm 4]$	183	2636
$W^{(k)}[\pm 5]$	19	1245
$W^{(k)}[\pm 6]$	1	498
$W^{(k)}[\pm 7]$		169
$W^{(k)}[\pm 8]$		48
$W^{(k)}[\pm 9]$		12
$W^{(k)}[\pm 10]$		2
$V^{(k)}$	2.00	6.00