Genome Assembly with Long Noisy Reads

Amir Kadivar

April 2016

abstract *We consider the problem of de novo genome assembly with reads characterized by relatively large length and high error rate. We propose an algorithm within the overlap-layout-consensus framework that can efficiently detect and align overlapping reads. Our approach relies on statistical properties of k-mers (aka l-tuples or words) that emerge from long sequencing reads and provides a statistical test that can quickly distinguish between overlapping and non-overlapping sequences with high power. Furthermore, the algorithm provides an estimate of the relative offset between overlapping sequences which can be used to solve a banded alignment problem. Mathematical analysis of various components of the algorithm, its performance on real data, and directions for future work are presented.*

Contents

Background

Second generation sequencing technologies (e.g. Illumina, 454) are characterized by relatively short reads (on average 100-500 bps), low error rates, and under 10X coverage with the main challenge of de novo assembly being the resolution of repeat regions. Third generation sequencing technologies (e.g. Pacific Biosciences SMRT, Oxford Nanopore MinION) are characterized by longer reads (on average 5-15 kbps), high error rates (up to %20), and up to 40X coverage. Therefore, single-molecule sequencing drastically simplifies the problem posed by repetitive structures. However, established

assembly schemes do not scale well to the dimensions and accuracy levels of such technologies: overlap-layout-consensus (OLC) schemes suffer due to the high error rates (specifically high indel rates) and de-Bruijn graph schemes suffer from large read lengths.

There have been successful attempts at incorporating SMRT reads into assembly pipelines virtually all of which fall within the OLC framework. Most commonly, long reads are used in tandem with second generation techniques mainly in the finishing process and to resolve repetitive structures [1]. Here we are concerned with the problem of assembly using *only* SMRT reads. Proposed solutions for this problem include HGAP [2], MHAP [3], and SPARC [4].

Terminology and scope

The *de novo* genome assembly problem is distinguished from a corresponding, rather easier problem of *mapping* reads to an existing reference genome. The discussion that follows applies to both cases equally while our focus is on the former. For *de novo* genome assembly, within the OLC framework, we first seek pairwise alignments between the reads and then hope to somehow combine them into a multiple sequence alignment which will then be used to recover a *consensus* sequence for the entire genome. The natural choice of alignment between reads is what we will refer to as *overlap* alignment (or a suffix-prefix alignment).

Given a sequence of reads $(R_n)_{n=1}^N$ we wish to find the *overlap graph* $G = (V, E)$ which is a *weighted directed acyclic* graph whose vertices V is $\{R_n\}_{n=1}^N$. Two reads R_i and R_j are overlapping with score w, denoted by $R_i \overset{w}{\mapsto} R_j$, if a suffix of R_i aligns with a prefix of R_j [1]. Our goal is to find an approximate $\hat{E}$ to the set E of all such edges and we define the sensitivity and specificity of our results in terms of the number of edges recovered from the *true* overlap graph (built using a known reference genome). That is, we wish to minimize both of:

$$f.n. = \frac{|E \setminus \hat{E}|}{|E|} \text{ , and } \quad f.p. = \frac{|\hat{E} \setminus E|}{|E|}$$

Our overlap discovery algorithm relies on k-mer methods. Namely, all analysis begins with indexing all words of length k observed in all reads. This provides, for any pair of reads, a list of *seeds*, which are exactly matching words together with their respective positions, in the two reads. We will propose that the statistical properties of these seeds can be used to significantly reduce the time spent on alignment. First, the distribution of seeds can be used to rule out possible overlaps *without* alignment. Second, when alignment is necessary the same statistical properties can be used to reduce the complexity of alignment from quadratic to linear time and space by using a *banded* variant of the alignment algorithm.

A *seed* z for reads R_i and R_j is a pair of exactly matching substrings of R_i and R_j. If the starting positions of the shared word is z_i and z_j in R_i and R_j, we call $z_i - z_j$ the *shift* of the seed z. We

[1] This is accurate as long as the alignment between the R_i and R_j is not substring. The accurate formulation is this: $R_i \overset{w}{\mapsto} R_j$ if the high-scoring alignment between the two reaches the boundary of both sequences and starts at the beginning of R_j.

will show that the distribution of seed shifts for a pair of reads can be used to **i)** rule out potential overlaps with marginal computational cost, and **i)** provide a hint for the diagonal band of the dynamic programming table that should be populated if the sequences are potentially overlapping.

The statistical property of interest is the distribution of shifts. The heuristic is that if two sequences are overlapping there must exist a narrow diagonal range in the dynamic programming table where there is a relatively high concentration of seeds. Note that finding this range does not require any alignment:

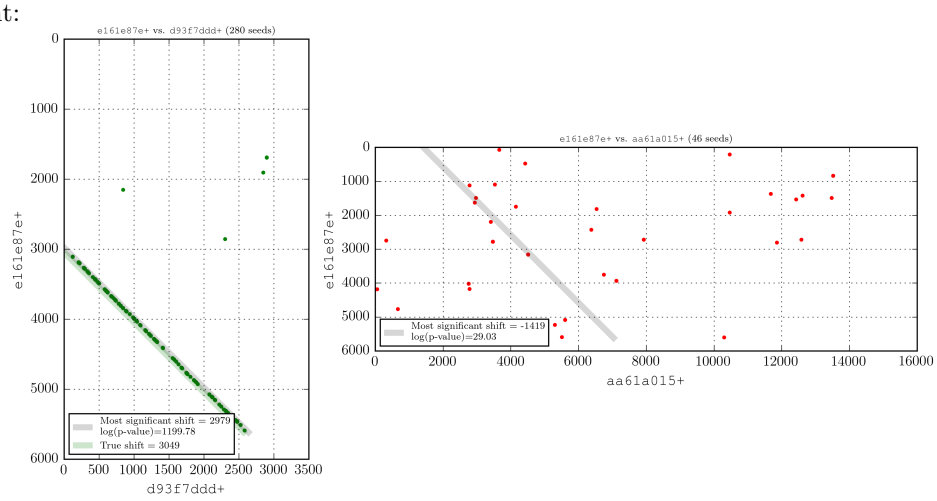


Figure 1: Example raster plots of seeds demonstrating the central idea of dense diagonal bands. *Left*: Seeds for a pair of overlapping sequences are plotted in the plane according to their staring positions in each read. The overlaid gray band is the most dense band as detected by our algorithm. The green band is calculated from the mappings of each read to a known reference genome using `blasr` (see below). *Right*: Same for a pair of non-overlapping sequences.

Overlap discovery algorithm

The algorithm proceeds as follows (see pseudo-code below):

1. Scan all occurrences of all words in all sequences.
2. Exclude all words that appear too often in the collection of reads.
3. For each pair of reads find a diagonal range with a significant number of seeds.
4. If such a range does not exist, rule out the possibility of overlap.
5. If such a range exists, perform a banded alignment in linear time over the found range. If the alignment has acceptable quality report the overlap.

In each step the following parameters are exposed to the user (analyzed in detail in further sections):

1. The length of k-mers.
2. Repetitive words are an attractive target for optimization since they are few in numbers but they appear in many sequences and lead to many seeds. Their exclusion is based on the

heuristic that they potentially are part of repeat regions. This requires a threshold for the significance of observing a given word a large number of times across reads.

3. Shift statistic calculations depend on two global parameters: g, ϵ. The former is the probability of a gap introduced at an arbitrary position by the sequencing machine and the latter is a parameter internal to the probabilistic model (see analysis below and appendix I for details).
4. Ruling out non-overlapping sequences requires a minimum *threshold for significance* of the number of seeds observed in a diagonal range.
5. Banded alignments are terminated prior to termination according to a *stop criterion* if they appear to not be leading to high scores. Completed alignments are examined based on multiple *acceptance criteria* which exclude barely-overlapping and mostly-overlapping pairs of sequences (defined below).

Detecting dense diagonal bands

For each pair of reads (R_i, R_j), overlapping or not, we first seek to find the shift range with the highest density of seeds (i.e number of seeds divided by shift range area).

Consider the distribution of seeds in the (z_i, z_j) plane where z_i, z_j are the starting coordinates of each seed. In this plane, all seeds reside within the rectangle $M = [0, |R_i|] \times [0, |R_j|]$. We assume that under the null hypothesis (the two sequences are not overlapping) seeds are uniformly distributed in M. Further, in this plane a shift window $[d - r, d + r]$ corresponds to a trapezoidal strip constrained between two main quadrant diagonals centered around the d diagonal and at anti-diagonal distance r of one another. A seed (z_i, z_j) lies within the strip if $|z_i - z_j - d| < r$. For any shift d, suppose we observe n seeds within the shift range $[d - r, d + r]$. The p-value under the null hypothesis is:

$$\Pr(X_d \geq n) \simeq \left(\frac{A_d}{|R_i| \cdot |R_j|} \right)^n$$

where A_d is the area of the trapezoidal strip corresponding to shift d:

$$A_d \simeq 2r \sqrt{(|R_i| - |d|)^2 + (|R_j| - |d|)^2}$$

The final *significance* value we use to find the most significant shift is:

$$\sigma(d) = \log(|R_i| + |R_j|) + n \left[\log(A_d) - \log(|R_i|) - \log(|R_j|) \right]$$

where the first term is a Bonferroni correction since the process involves testing the same hypothesis for $|R_i| + |R_j|$ different values of d.

As the above significance formula indicates, each seed has a constant additive effect on the total significance of all bands it appears in. Therefore, with a justifiable approximation, the significance values can be collected in one-pass as seeds are found between each read-pair (see pseudo-code below) by assuming that each seed z contributes to those bands whose center shift d' lies in the corresponding band centered at z. In other words, z contributes to $\sigma(d)$ for all d in the set:

$$\{d : |d - d_z| < r(d_z)\}$$

where $d_z = z_i - z_j$ is the shift of z and $r(d)$ assigns a band radius to each shift.

Choice of diagonal band width

As a consequence of the results in appendix I, this parameter can be absorbed in the probabilistic model used for banded alignment: the same r^* used for the banded alignment can be used for detecting bands dense with seeds. As the analysis below makes clear, the width of a diagonal band is obtained such that there is a small probability for an alignment with assumed gap probabilities which starts at the center of the band to escape it by its end. As we will see, a fixed choice of width sacrifices too much discriminatory power and thus, the diagonal band width should be calculated for each read-pair and at each shift value. In the analysis of appendix I, it is shown that the exact analytic formula of the probabilistic model are too inefficient for this approach. Instead, approximate schemes (with bound error) are presented that effectively remove the overhead of calculating the band radius at every shift for every read-pair.

Banded alignment

A banded alignment problem with width B is one where only those alignments are considered that entirely lie within a range of diagonals in the DP table with maximum anti-diagonal distance B. For a pair of sequences with length $O(n)$ both time and space complexity of a *global* banded alignment is $O(Bn)$ instead of the usual $O(n^2)$ (see appendix IV). Some considerations are due for applying banded alignments to the overlap discovery problem.

1 Applicability to overlap discovery

Banded *local* alignments do not reduce the complexity since the band constraint is with respect to the starting position of the alignment and the entire DP table must be populated regardless. The same applies to the general overlap discovery problem. However, if we limit the search to those alignments confined to a diagonal range (for example, the shift range proposed by a statistical analysis of shifts) time and space complexity are reduced to $O(Bn)$ where $n = \min(|S|, |T|)$. This means banded overlap alignment is suitable for both overlap discovery in *de novo* assembly and mapping reads to existing reference genomes.

2 Choice of band radius

Consider two sequences S and T. The range of possible shifts is: $\{-|T|, \ldots, 0, \ldots, |S|\}$. Clearly the optimal band radius varies as the starting shift sweeps the range of possible values, with starting shifts closer to zero requiring a larger radius.

We formulate the problem as follows: we wish to find the smallest band radius r^* such that the probability ϵ of missing the true alignment due to it leaving the enforced band is a given fixed value

(say 10^{-3}). The calculations in appendix I provide the following approximation [2]:

$$r \geq 2\text{erf}^{-1}(1-\epsilon)\sqrt{g(1-g)K}$$

where g is the gap probability and K is the expected length of the alignment. See appendix I for probabilistic analysis.

3 Choice of K

The quantity K in the above formula is the (unknown) length of an arbitrary alignment. The range of possible values of K is quite large: an all indel alignment would be roughly twice as long as an all matching alignment. It seems reasonable, but not yet justified, to use the following approximation for a typical high scoring alignment:

$$K \simeq \frac{2}{2-g}L$$

where g is the gap probability and L is the length of an all-matching alignment:

$$L(d) = min(|S| - d, |T|) + min(d, 0)$$

4 Stop Criterion

In order to terminate alignments which do not "seem promising" we keep track of the number of new global minimums in score encountered throughout the alignment and stop the extension as soon as the number of new minima encountered exceeds a threshold (e.g. 10). This is motivated by the observation that the progression of alignment score has very simple long range dynamics: for overlapping sequences the score continually increases and for non-overlapping sequences the score continually decreases.

5 Acceptance Criteria

A common category of f.p.'s is that of those caused by *mostly-overlapping* sequences. Two reads are mostly-overlapping if their correct overlap alignment starts and/or ends very close to the main diagonal of the dynamic programming table (e.g. 200 bp). In such cases the direction of the overlap is not robustly determined by the optimal alignment hence potentially reversing the direction of a heavy edge in the overlap graph. Due to the high sensitivity of the layout path to f.p.'s and the small information content of such read pairs, we choose to not add an edge (in either direction) to the graph if two reads are mostly-overlapping [3].

Similarly, *barely-overlapping* sequences are those that only overlap over a short (e.g. 200 bp) substring. Such read-pairs are also risky for two reasons. First, parameter ranges that ensure their

[2] Here is a typical value: for aligning two sequences $\geq$ 10 kbp in length, a band radius of 150 is enough to guarantee $\epsilon < 0.001$ under gap probability 0.1.

[3] An alternative would be to add both edges to the graph or to remove one of the reads entirely from the graph.

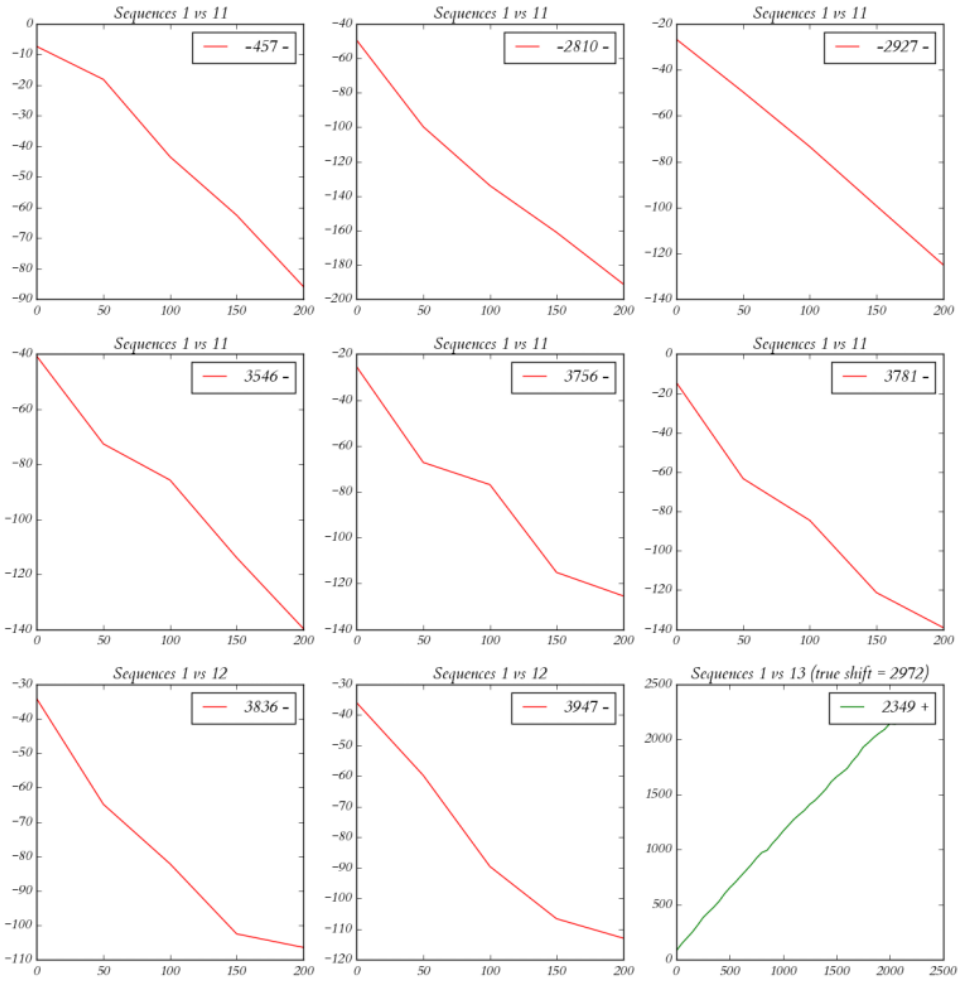


Figure 2: Progression of scores *throughout* the alignment of overlapping (*green*) and non-overlapping (*red*) read-pairs.

discovery necessarily misidentifies some non-overlapping read pairs as overlapping. Second, short overlaps may be artificially induced by repeat regions. By ignoring such short overlaps we circumvent the possibility of mistaking non-overlapping reads with similar shared repeat regions at their opposite endpoints (suffix of one and prefix of the other).

Finally, since alignment quality is the sole measure of ruling out non-overlapping read-pairs with high shift peaks some measure of alignment quality should be incorporated. In the results we present below percentage of exact matches (e.g. with cutoff %70) have been used as such a measure. Note that ideally this criterion and the "max. new min.s" criteria should be merged into one exposed parameter (see the section on future directions).

Repetitive words

Repeat regions of various kinds are a challenge in any eukaryotic de novo assembly. The challenge has two components:

- Possibility of mistaking various copies of a repeat region along the genome leading to false positive overlaps.
- Time wasted on trying to extend seeds coming from various copies of a repeat region.

A simple idea is to discard seeds that appear "too often" across the reads. Preliminary tests show improvements from discarding words with very small p-values (roughly e^{-5}). See appendix II for probabilistic analysis.

Implementation

All code is open source and available at github.com/amirkdv/biseqt and documentation as well as library API can be found at biseqt.readthedocs.org [4]. The dynamic programming algorithm for sequence alignment is implemented in C. All k-mer handling (B-tree indices, disk IO, querying, etc.) are delegated to SQLite which is a fast, serverless, SQL database implemented in C. All graph handling (cycle breaking, topological sorting of DAG's, and drawing graphs; not reported here) are delegated to igraph which is implemented in C/C++. Everything else is implemented in Python interfacing the C component via a foreign-function interface (relying on `cffi` from PyPI) and interfacing SQLite and igraph via their official python modules (`igraph` and `sqlite3` from PyPI, respectively).

Pseudo-code

Here is a Python-esque pseudo-code for the overlap discovery algorithm:

```
# Given two sequences tries to find an overlap alignment between them:
def discover_overlap(S, T):
```

[4] Note that under the currently heavy load of active development, the documentation remains slightly out of date.

```
    S_word_hits = scan(S) # a map from words to positions in S
    T_word_hits = scan(T) # a map from words to positions in T
    shift_scores = {}     # a map from diagonal number to score
    # seeds arise from words appearing in both S and T at least once:
    for word in S_word_hits.intersection(T_word_hits):
        if repetitive(word):
            continue
        # each pair of occurrences of a word in S and T defines a seed:
        for i, j in S_word_hits[word] × T_word_hits[word]:
            shift = i - j
            # all neighboring diagonals (within distance r) get a score
            # contribution s from seed (i,j):
            r = radius(|S|, |T|, shift)
            for k in range(shift - r, shift + r):
                shift_scores[shift] += seed_contribution(|S|, |T|, shift)

    if max(shift_scores) > min_score:
        # max_shift is the shift that accumulated the highest score
        r = radius(|S|, |T|, max_shift)
        return align(S, T, between=(max_shift - r, max_shift + r))
```

Correct labels

To test the algorithm we rely on a correct labeling of read pairs as overlapping and non-overlapping. This requires mapping the reads to a reference genome using another tool and labeling read pairs according to their mapped coordinates. Three alternatives, `bwa` [5], `blasr` [6], and `lastz` were used under their default parameters and under adjusted parameters to make sure all programs are comparing sequences equally. Under both cases, the mappings of `blasr` and `bwa` were more consistent with each other than with `lastz`.

Results and discussion

In all the following results 1055 Pac Bio reads corresponding to chromosome 1 of *Leishmania Donovani* are provided to the algorithm. Correct mappings are obtained by `blasr` and the correct decision of overlap vs non-overlap is obtained from these mappings. Of all the roughly 2 million read-pairs to be considered (including reverse complements), only about %2 are truly overlapping. The first measure of the success for our algorithm is its discriminatory power based only on shift statistics. Since not all read-pairs can be distinguished solely on this basis, a second measure of accuracy is the f.p. and f.n. rates over the assembled overlap graph as compared to the graph obtained from mapping to a reference genome by `blasr`.

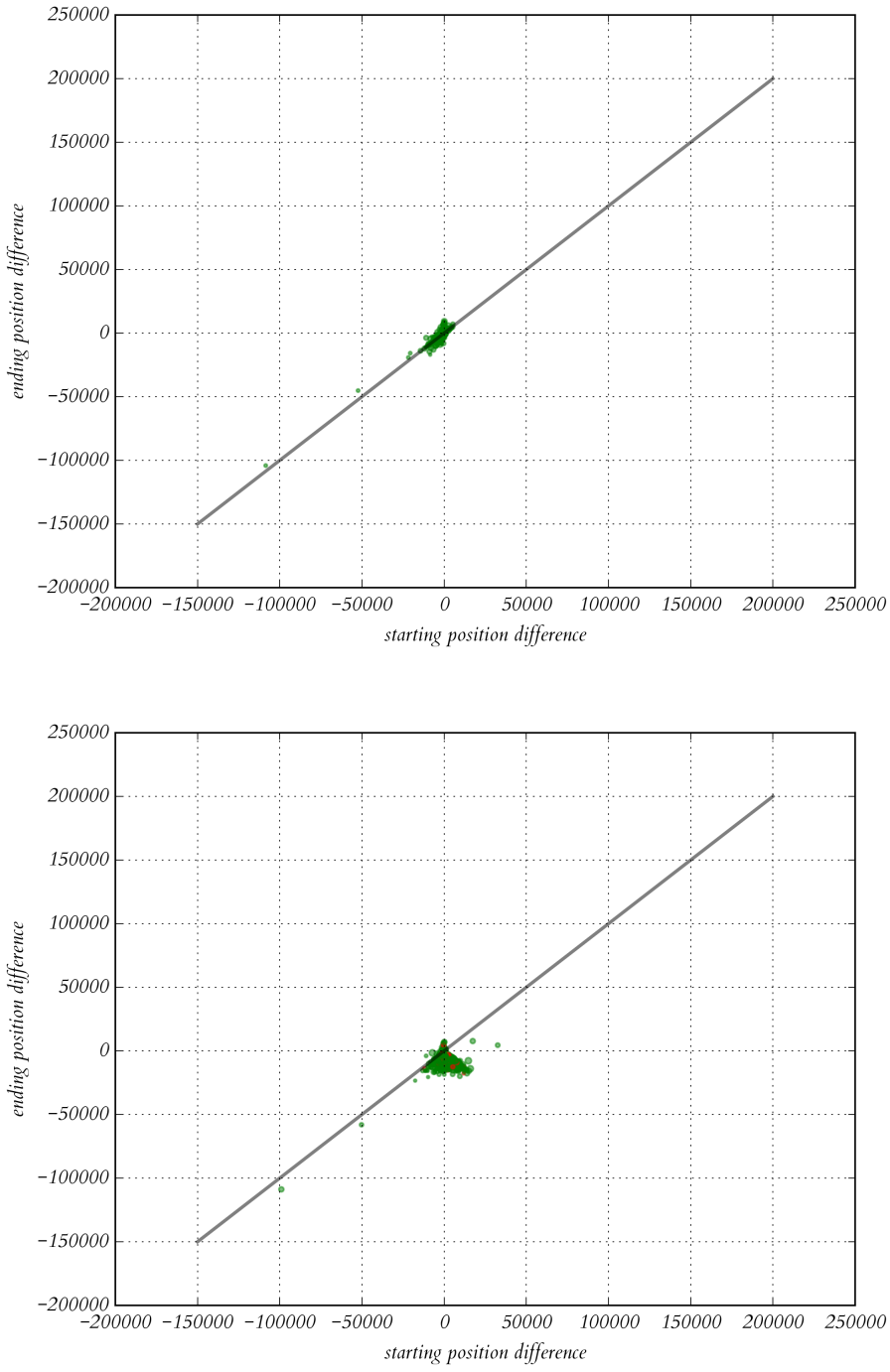


Figure 3: The agreement between `blasr`, `bwa`, and `lastz` over Pac Bio reads corresponding to chromosome 1 of *Leishmenia Donovani*. In each plot green dots represent reads mapped to the same strand by each tool and the coordinates represent the distance in mapped positions. Red dots represent reads mapped to opposite strands by the two tools. *Top*: comparison of mappings by `bwa` and `blasr`. *Bottom*: comparison of mappings by `lastz` and `blasr`.

Discrimination by shift statistics

Shift statistics are capable of discarding the overwhelming majority of non-overlapping read pairs based on their lack of a dense diagonal band. Fig. 4 summarizes the discriminatory power of our algorithm: with appropriate choice of cutoff, %99 of non-overlapping read-pairs can be discarded while only misidentifying %5 of overlapping read-pairs. The entire operation for the *Leishmania* dataset (from scanning words to finding the most dense band for each read-pair) completes in about 1 hour on a personal computer [5] with less than 3 GB of space [6].

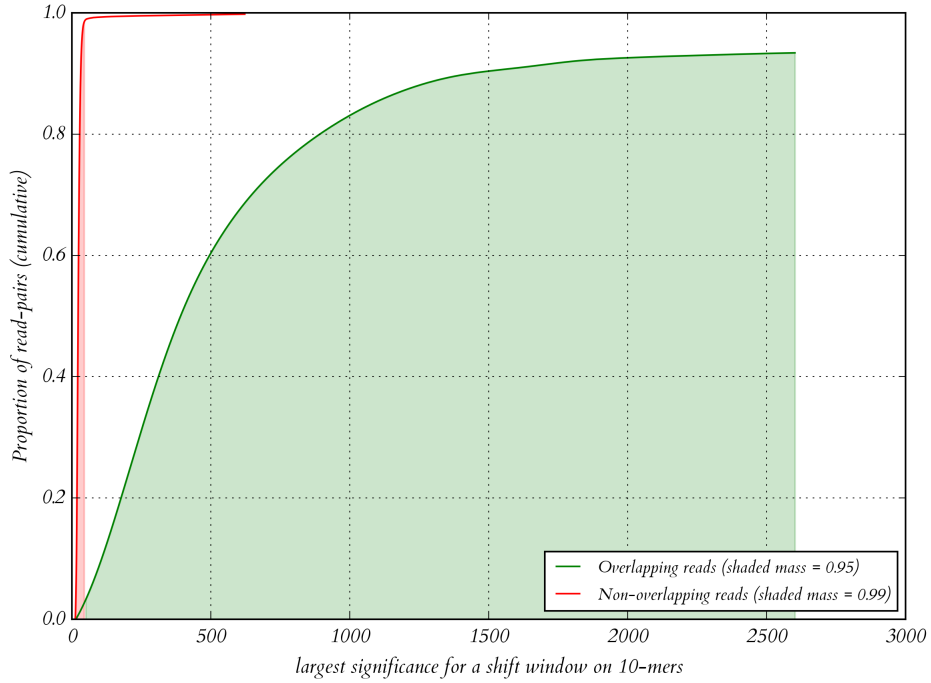


Figure 4: Cumulative distribution of the significance $\sigma(d)$ of the most dense diagonal band for Pac Bio read-pairs corresponding to chromosome 1 of *Leishmenia Donovani*. The red curve is the cumulative distribution of non-overlapping reads (curve is that of overlapping read-pairs. All barely-overlapping reads (<500bp) are excluded. The shaded regions show the discarded non-overlapping pairs and the leftover overlapping pairs after applying a significance cutoff of $\sigma > 50$ for overlapping pairs.

Banded alignments

A banded alignment following the shift statistic analysis is necessary for two reasons. First, as seen above, shift statistic alone is not enough to identify all overlapping pairs. Second, depending on the protocol of communication between the overlap-detection phase and the rest of the assembly pipeline, a more accurate account of overlaps (e.g. alignments) are needed (however, see section

[5] `Ubuntu 14.04` with an Intel quad-core `3.67 Hz` CPU.

[6] The majority of space consumption is in external memory, i.e shuffled back and forth to disk. RAM usage never exceeds `1 GB`.

on future directions). Figure 5 shows an example of successful completion of the algorithm in two phases: first, a dense diagonal band is identified; then, a banded alignment confirms the overlap and provides an edit transcript.

Over the entire dataset, performing banded alignments over the chosen read-pairs (roughly 50,000) takes about 4.5 hrs on a personal computer [7]. However, f.n and f.p. rates were surprisingly high: %45 and %10, respectively (see below).

Future directions

Mapping to reference

The current results were obtained by comparing to `blasr` mappings. Similar results were obtained when comparing to correct overlaps obtained by comparing to `bwa` mappings. However, there remain crucial questions about the quality of "true" overlap labels in the data set. First, despite rough agreement between the two tools, which are standards for aligning Pac Bio reads [8], they differ considerably on position: roughly %30 of reads are mapped to positions farther than 1.5 kbp of each other by `blasr` and `bwa`. This is a large margin compared to average read lengths (5-15 kbp) and can affect much of the results. Second, a true test of the viability of our algorithm is to compare it to `blasr` and `bwa` for the simpler problem of mapping reads to a reference. This has been implemented currently yielding wildly different mappings. This and the unexpectedly low performance of banded alignment outputs presented above possibly hint towards an implementation bug that needs to be investigated. Finally, the filtering of chromosome 1 reads from the larger collection of sequencing output may contain errors of its own [9].

Overlap discovery: accuracy

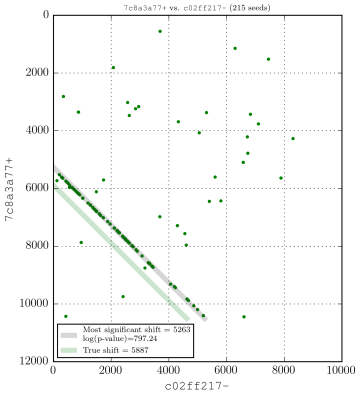
Aside from the possibility of corruption in correct labels, potential offenders for low accuracy are:

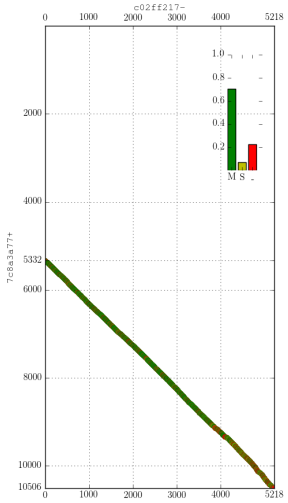
- As mentioned above, implementation bugs.
- Due to the slow pace of obtaining results on a personal computer, all parameters where fixed to rough guesses obtained from experiments on smaller portions of the data set. These include parameters highlighted above and the parameters of the alignment (and implicit assumptions of gap probabilities by the sequencing machine). A series of parameter-sweeping runs are needed to explore the possibility of improvements by better choice of parameters.
- Both `blasr` and `bwa` perform soft clipping of the reads which complicates calculating the true mapping for the entire read (our algorithm does not perform soft clipping). Currently, the starting position is simply pulled back by the size of soft clipping. However, it remains unclear

[7] System spec and space consumption profile the same as before.

[8] `blasr` is developed and maintained by the Pacific Biosciences team and `bwa` [5] has been used as baseline in other studies [2], [3] and has recently introduced a a default parameter regime tuned for Pac Bio reads (usage via `bwa mem -x pacbio` ...).

[9] In fact, both `blasr` and `bwa` report low quality on a number of reads and `blasr` refuses to map 10 of the reads.





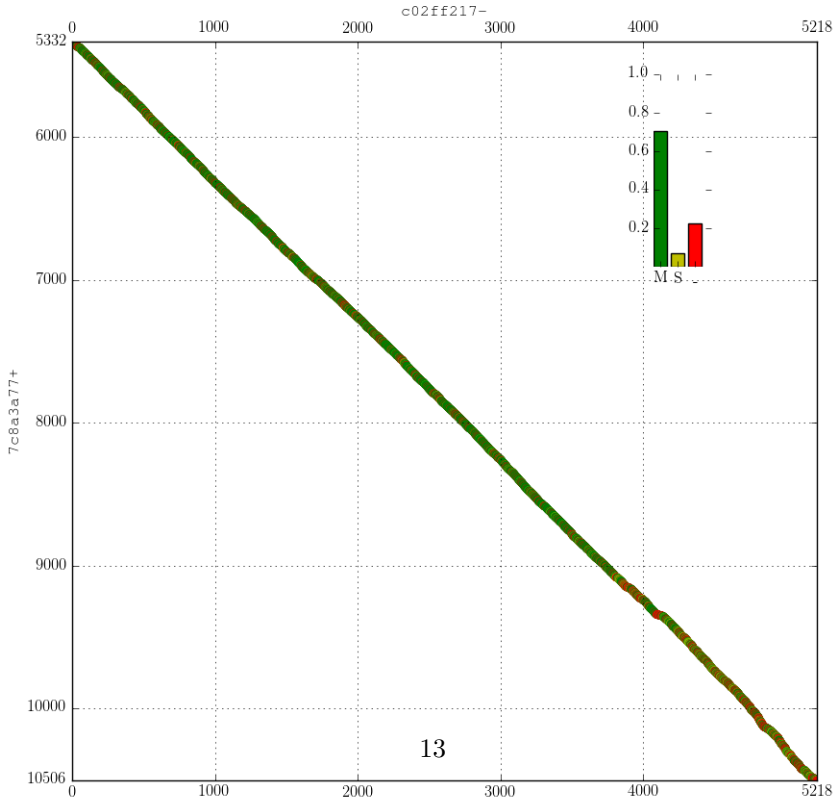


Figure 5: Example of successful completion of the algorithm. *Top left* The seed rasterplot for two

what the effects of this are, and if they are in any way related to often-seen anomalous cases where the estimated shift between supposedly overlapping reads differs significantly with the location of a dense diagonal band (cf. Fig. 6). Furthermore, the reference genome used to produce the results above contains many ambiguous regions which are in turn treated ambiguously by `blasr` and `bwa` (e.g the latter replaces any `N` with a random nucleotide). The effects of ambiguous regions has not yet been investigated.

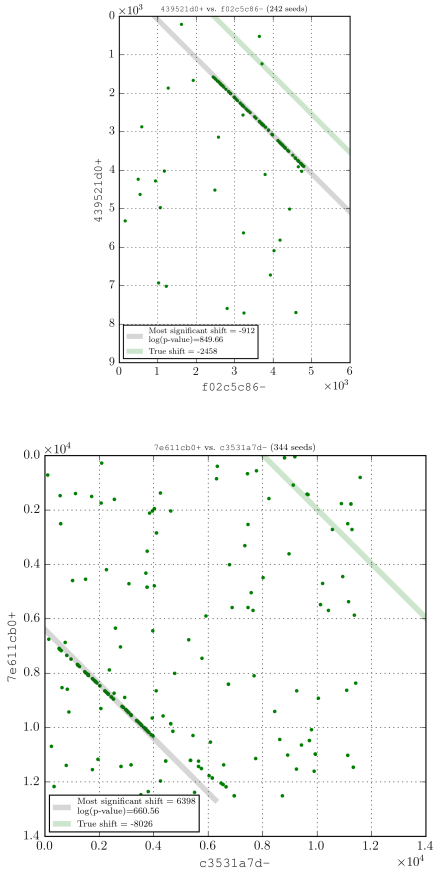


Figure 6: Examples of anomalous cases where the estimated shift between overlapping reads derived from mapping to reference by 'blasr' or 'bwa' is significantly different from the most dense diagonal band. As before, the gray band is detected by our algorithm and the green band is what mapping by reference implies.

Overlap discovery: speed

Although preliminary results are acceptable and the algorithm and implementation is highly parallelizable, there is potentially room for significant performance improvements. First, despite the light weight and comparably stellar performance of SQLite among SQL databsases, it is possible that better suited external memory schemes may yield high performance improvements [10]. Second, although our dynamic programming algorithm is implemented in C, shift statistic calculations are performed in Python for convenience of IO operations and communication with SQLite. Profiling

[10] An alternative is `HDF` (which is used by `blasr` internally).

these calculations and identifying hot loops that can be moved to C is needed.

Consensus sequence

Known approaches for incorporating pairwise alignments of long noisy reads into a consensus sequence include HGAP [2], T-Coffee [7], and POA [8], [9]; with all of which our algorithm can be, in full or in part, integrated. There are, however, alternative heuristics based on the strength and weaknesses of our algorithm. For instance, suppose once a reasonable $\hat{E}$ is found we proceed as follows: **1.** find a *layout path* by solving a heaviest path problem over the overlap DAG [11]., **2.** build a "scaffold" by solving a small consensus problem over the above layout path, **3.** align all vertices that were left out of the layout path onto the scaffold and return a complete layout path, **4.** solve the consensus problem as usual. Alternatively, the shift estimates can be used to divide the unknown genome into regions where it is roughly known, without alignment, which parts of which reads will contribute. Then multiple MSA problems can be solved over each such regions.

Chainable seeds

When counting all seeds that lie within a diagonal strip we are ignoring the fact that not all such seeds can belong to the same alignment. Furthermore, it could be possible to recover the overall alignment simply by chaining the seeds, namely by only solving the DP algorithm over the distances between "chainable" seeds. The idea of chaining has already been discussed in the literature [10], [11]. This can be achieved by organizing the seeds from a diagonal band into a DAG where each seed is connected to its closest chainable neighbors. Appendix III provides a preliminary probabilistic model for discrimination based on the distance between subsequent seeds in such a chain.

[11] Note that our overlap graph is weighted and directed differently than the usual OLC graph which requirres a Hamiltonian path for finding the layout path.

References

[1] S. Koren, M. C. Schatz, B. P. Walenz, J. Martin, J. T. Howard, G. Ganapathy, Z. Wang, D. A. Rasko, W. R. McCombie, E. D. Jarvis, and A. M. Phillippy, "Hybrid error correction and de novo assembly of single-molecule sequencing reads," *Nat Biotech*, vol. 30, no. 7, pp. 693–700, 2012. DOI: 10.1038/nbt.2280.

[2] C.-S. Chin, D. H. Alexander, P. Marks, A. A. Klammer, J. Drake, C. Heiner, A. Clum, A. Copeland, J. Huddleston, E. E. Eichler, S. W. Turner, and J. Korlach, "Nonhybrid, finished microbial genome assemblies from long-read smrt sequencing data," *Nat Meth*, vol. 10, no. 6, pp. 563–569, 2013, Article.

[3] K. Berlin, S. Koren, C.-S. Chin, J. P. Drake, J. M. Landolin, and A. M. Phillippy, "Assembling large genomes with single-molecule sequencing and locality-sensitive hashing," *Nat Biotech*, vol. 33, no. 6, pp. 623–630, 2015, Research.

[4] C. Ye and S. Ma, "Sparc: A sparsity-based consensus algorithm for long erroneous sequencing reads," PeerJ PrePrints, Tech. Rep., 2015.

[5] H. Li and R. Durbin, "Fast and accurate short read alignment with burrows–wheeler transform," *Bioinformatics*, vol. 25, no. 14, pp. 1754–1760, 2009.

[6] M. J. Chaisson and G. Tesler, "Mapping single molecule sequencing reads using basic local alignment with successive refinement (blasr): Application and theory," *BMC bioinformatics*, vol. 13, no. 1, p. 238, 2012.

[7] C. Notredame, D. G. Higgins, and J. Heringa, "T-coffee: A novel method for fast and accurate multiple sequence alignment," *Journal of molecular biology*, vol. 302, no. 1, pp. 205–217, 2000.

[8] C. Lee, C. Grasso, and M. F. Sharlow, "Multiple sequence alignment using partial order graphs," *Bioinformatics*, vol. 18, no. 3, pp. 452–464, 2002.

[9] C. Grasso and C. Lee, "Combining partial order alignment and progressive multiple sequence alignment increases alignment speed and scalability to very large alignment problems," *Bioinformatics*, vol. 20, no. 10, pp. 1546–1556, 2004.

[10] M. Brudno, M. Chapman, B. Göttgens, S. Batzoglou, and B. Morgenstern, "Fast and sensitive multiple alignment of large genomic sequences," *BMC bioinformatics*, vol. 4, no. 1, p. 66, 2003.

[11] L. Noé and G. Kucherov, "Improved hit criteria for dna local alignment," *BMC bioinformatics*, vol. 5, no. 1, p. 149, 2004.

[12] W. Feller, *An introduction to probability theory and its applications*, ser. Wiley series in probability and mathematical statistics: Probability and mathematical statistics. Wiley, 1971.

[13] S. Aki, H. Kuboki, and K. Hirano, "On discrete distributions of order k," *Annals of the Institute of Statistical Mathematics*, vol. 36, no. 1, pp. 431–440, 1984.

[14] A. N. Philippou, C. Georghiou, and G. N. Philippou, "A generalized geometric distribution and some of its properties," *Statistics & Probability Letters*, vol. 1, no. 4, pp. 171–175, 1983.

[15] M. J. Barry and A. J. L. Bello, "The moment generating function of the geometric distribution of order k," *The Fibonacci Quarterly*, vol. 31, pp. 178–180, 1993.

Appendices

I Statistics of seed shifts

Let R_i and R_j be any two reads and let them have a set of n seeds $\{z^k\}_{k=1}^n$ where seed z^k has coordinates (z_i^k, z_j^k). Let the shifts of the seeds be $\{d_k\}_{k=1}^n$ where

$$d_k = z_i^k - z_j^k$$

Each seed, through its coordinates, implies a certain offset for the correct overlap alignment of R_i and R_j. The central idea is that if R_i and R_j are actually overlapping reads, there must be concentration of shift values. That is, if one looks at the histogram of $\{d_k\}_{k=1}^n$ there is a certain "peakedness" close to the correct shift between the sequences. In fact, one can visually verify that this is typically the case [12].

Consider an alignment of length K. The succession of anti-diagonal distances (from the starting diagonal) along the cells of the alignment gives rise to a 1d random walk over $\mathbb{Z}$. For the random walk, the probability of $+1$ and -1 moves are both p (corresponding to an indel in the alignment) and the probability of no move (corresponding to a match or substitution) is $1 - 2p$.

Define $u(n, k)$ to be the probability of such a random walk starting at position n (i.e starting shift n) to be within a band of radius r at time k (i.e after k steps of the alignment). The recurrence relation for u is:

$$u(n, k) = pu(n - 1, k - 1) + pu(n + 1, k - 1) + (1 - 2p)u(n, k - 1)$$

subject to boundary condition $u(n, k) = 0$ for any $n > r$ at all times k. Our goal is to find a closed form and invertible solution to $u(n, k)$ which would be used to find r such that $u(0, K) > 1 - \epsilon$ for given K and ϵ.

Generating Functions

There are two potential *formal* power series to consider for solving for $u(n, k)$. Neither lead to a usable solution. The first alternative is the following:

$$f_n(x) = \sum_k u(n, k)x^k$$

Substituting the series into the recurrence relation leads to the recurrence $f_{n+1} = gf_n - f_{n-1}$, where $g = 1/px + 2 - 1/p$, which is no easier than the original problem. The second alternative is:

$$f_k(x) = \sum_n u(n, k)x^n$$

which then gives the solution:

$$f_k(x) = \left(px + (1 - 2p) + \frac{p}{x}\right)^k f_0(x)$$

[12] The most common exception is when two reads are only weakly overlapping, i.e they overlap on a short substring, which is neither surprising nor problematic.

where:

$$f_0(x) = \sum_{i=-r}^{r} x^i$$

This is again a recurrence relation that has to be solved in quadratic time for every round of band calculation (this is precisely what [11] does) which is unacceptable for the size of the genome assembly problem [13].

Diffusion equation

We can rearrange the recurrence relation into the following form:

$$u(n,k) - u(n,k-1) = p\left[u(n-1,k-1) + pu(n+1,k-1) - 2u(n,k-1)\right]$$

which is precisely the discretization of the 1D diffusion equation $u_t = pu_{xx}$. We can now argue that $u(n,k)$ can be approximated by the analytic closed form solution of the diffusion equation, subject to corresponding boundary conditions. This would be valid if the discretization above is numerically stable. The stability criterion for the finite difference approximation of the diffusion equation is:

$$p \leq \frac{(\Delta x)^2}{2\Delta t}$$

In our case we have $\Delta x = \Delta t = 1$ and thus the stability criterion is $p \leq 1/2$ which is true since $1 - 2p \geq 0$.

Numerics

The continuous diffusion equation corresponding to our recurrence relation is the following IBVP:

$$\begin{cases} u_t = pu_{xx} \\ u(r,t) = u(-r,t) = 0 \\ u(x,0) = g(x) \end{cases}$$

where $g(x)$ is the unit impulse function centered at 0 with radius r, i.e $g(x) = 0$ if $|x| > r$ and $g(x) = 1$ otherwise.

Recall that in the end, our goal is to find the smallest value r^* such that the solution $u(x,t)$ of the above system for $r = r^*$ satisfies:

$$u(0,K) \geq 1 - \epsilon$$

for known ϵ (sensitivity parameter) and K ("expected" alignment length calculated from the shift d and the sequence lengths $|S|$ and $|T|$).

[13] Generally when using formal power series to solve recurrence relations, one hopes to find an analytic closed form for resulting power series and use McLaurin expansion to make it a "concrete", rather than formal, power series. But here our solution involves another recurrence relation *and* it involves negative powers of x which makes McLaurin expansion irrelevant.

If the boundary condition was not there, we could have solved the IVP problem using the Fourier transform and arrive at a nice analytic solution involving the erf function whose inverse is (numerically) known. We will return to this later. In the presence of the boundary constraint, however, the Fourier transform is inapplicable and the separation of variables algorithm gives the solution in the form of a generalized Fourier series. The main steps of calculations are listed below but note that a series solution is undesirable since the amount of calculation needed to find its inverse is unacceptable [14].

- We wish to solve the eigenvalue problem for $-\partial/\partial t$ and $-\partial^2/\partial x^2$ under the given boundary conditions, i.e we seek common eigenvalues λ_n and eigenfunctions X_n and T_n.
- The spatial eigenvalues are:

$$\lambda'_n = \left(\frac{n\pi}{r}\right)^2, \quad \lambda_n = \left(\frac{(n+\frac{1}{2})\pi}{r}\right)^2$$

  with eigenfunctions. $X_n = \cos(\sqrt{\lambda_n}x)$ and $X'_n = \sin(\sqrt{\lambda'_n}x)$. The differential operator $-\partial^2/\partial x^2$ can be checked to be *symmetric* with respect to the boundary conditions and therefore the spatial eigenfunctions are necessarily orthogonal.
- The general solution is therefore found by finding (a_n) and (b_n) such that:

$$u(x,t) = \sum_{n=0}^{\infty} \exp(-p\lambda_n t) X_n + \sum_{n=1}^{\infty} \exp(-p\lambda'_n t) X'_n$$

  satisfies $u(x,0) = g(x)$. It follows that (a_n) and (b_n) are the coefficients of the generalized Fourier series of $g(x)$ in the basis of the orthogonal eigenfunctions $\bigcup_{n\in\mathbb{N}}\{X_n, X'_n\}$. This gives:

$$a_n = \frac{2(-1)^n}{(n+\frac{1}{2})\pi}, \quad b_n = \frac{2(-1)^{n+1}}{n\pi}$$

The quantity we wish to bound is:

$$u(0,K) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)\pi} \exp\left(-pK\frac{(n+\frac{1}{2})\pi}{r}\right)$$

We now seek an approximation to this which is faster to compute than root finding on a series.

Proposition

(Approximate Problem): Let $\hat{P}_r$ be the following IVP obtained by relaxing the boundary condition of P_r:

$$\begin{cases} \mathscr{L}\hat{u}_t & = 0 \\ \hat{u}(x,0) & = 1 \text{ over } (-r,r) \end{cases}$$

Define r^* to be:

$$r^* \equiv 2\sqrt{pT}\,\mathrm{erf}^{-1}(1-\epsilon)$$

[14] Recall that every time any shift of any pair of seeds is considered, either for seed distribution or banded alignment, we need to calculate an appropriate radius.

Then the solution $\hat{u}$ of $\hat{P}_{r^*}$ satisfies:

$$\hat{u}(0,T) \geq 1-\epsilon$$

proof: For $\hat{P}_r$ Fourier transform would be applicable yielding the solution: $\hat{u} = \Phi * \hat{u}(x,0)$ where the convolution is over x and Φ is the diffusion kernel:

$$\Phi(x,t) = \frac{1}{\sqrt{4\pi pt}} \exp(-\frac{x^2}{4pt})$$

At any time t, this is precisely the probability density function of $\mathcal{N}(0, \sqrt{2pt})$ with cumulative distribution:

$$F(x;t) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x}{2\sqrt{pt}}\right)\right]$$

Thus:

$$\hat{u}(x,t) = F(x+r;t) - F(x-r;t) = \frac{1}{2}\left[\text{erf}\left(\frac{x+r}{2\sqrt{pt}}\right) - \text{erf}\left(\frac{x-r}{2\sqrt{pt}}\right)\right]$$

implying:

$$\hat{u}(0,T) = \text{erf}\left(\frac{r}{2\sqrt{pT}}\right)$$

Noting that erf is a monotonically increasing function completes the proof. ■

Remark

The formula for $\hat{u}(x,t)$ simplifies at $x = 0$ and at $x = \pm r$:

$$\hat{u}(0,t) = \text{erf}\left(\frac{r}{2\sqrt{pt}}\right) \quad \text{and} \quad \hat{u}(\pm r,t) = \frac{1}{2}\text{erf}\left(\frac{r}{\sqrt{pt}}\right) \tag{1}$$

Proposition

(Accuracy of approximation): Let r^* be such that the solution $\hat{u}(x,t)$ of the approximate IVP $\hat{P}_{r^*}$ satisfies:

$$\hat{u}(0,T) \geq 1-\epsilon$$

Then the solution $u(x,t)$ of the exact IBVP P_{r^*} satisfies:

$$u(0,T) \geq 1 - \frac{3}{2}\epsilon$$

proof: Define the residue $v(x,t) = \hat{u}(x,t) - u(x,t)$. Clearly, we have $v(x,t) \geq 0$ for all x and $t > 0$ and we wish to show:

$$v(0,T) \leq \frac{1}{2}\epsilon$$

We know that v satisfies the IBVP:

$$\begin{cases} \mathscr{L}v & = 0 \\ v(x,0) & = 0 \text{ over } (-r,r) \\ v(\pm r,t) & = f(t) \end{cases}$$

where, using (1), we have defined:

$$f(t) \equiv \hat{u}(\pm r, t) = \frac{1}{2}\text{erf}\left(\frac{r}{\sqrt{pt}}\right)$$

with $f(0) = 1/2$ to maintain continuity. Now define $h(x,t)$ for $t \geq 0$ by

$$h(x,t) = f(t) - v(x,t)$$

The function h satisfies the IBVP:

$$\begin{cases} \mathscr{L}h & = \dot{f} \\ h(x,0) & = \frac{1}{2} \text{ over } (-r,r) \\ h(\pm r,t) & = 0 \end{cases}$$

The solution is the sum of a particular solution and a homogenous solution:

$$h(x,t) = \frac{1}{2}\hat{u}(x,t) + [\Phi * \dot{f}]_{x,t}$$

where the convolution is over both x and t:

$$[\Phi * \dot{f}]_{x,t} = \int_0^t \int_{-\infty}^{\infty} \Phi(x-y, t-s)\dot{f}(s)dyds$$

Therefore:

$$\begin{aligned} h(0,T) &= \frac{1}{2}\hat{u}(0,T) + \int_0^T \dot{f}(s) \int_{-\infty}^{\infty} \Phi(y, T-s)dyds \\ &= \frac{1}{2}\hat{u}(0,T) + \int_0^T \dot{f}(s)ds = \frac{1}{2}\hat{u}(0,T) + f(T) - \frac{1}{2} \end{aligned}$$

where in the last step we have used the fact that $\Phi(\cdot, t)$ is a probability distribution over $\mathbb{R}$ for any t. Finally:

$$v(0,T) = f(T) - h(0,T) = \frac{1}{2}(1 - \hat{u}(0,T)) \leq \frac{1}{2}\epsilon$$

■

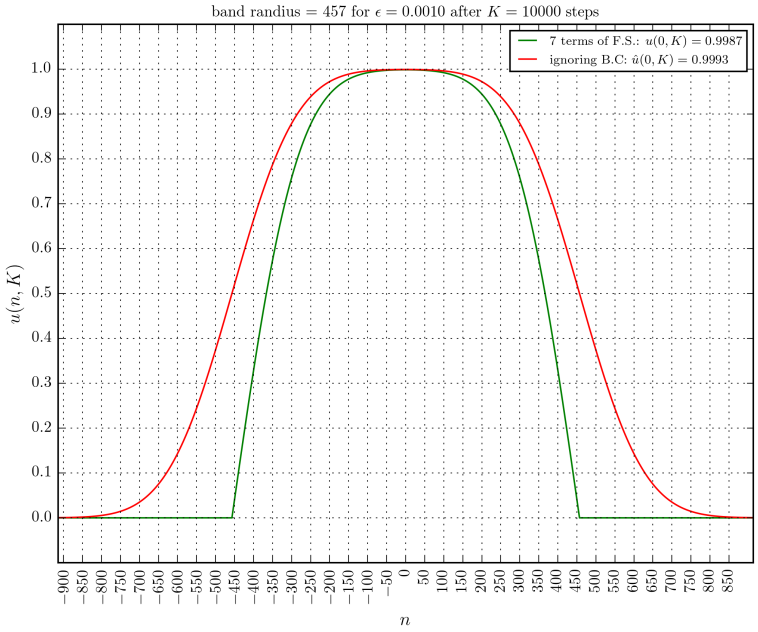
Corollary

(Approximation Algorithm): It follows that if r^* is defined as:

$$r^* \equiv 2\sqrt{pT}\text{erf}^{-1}\left(1 - \frac{2\epsilon}{3}\right)$$

then the solution u of P_{r^*} satisfies:

$$u(0,T) \geq 1 - \epsilon$$



II Repetitive words

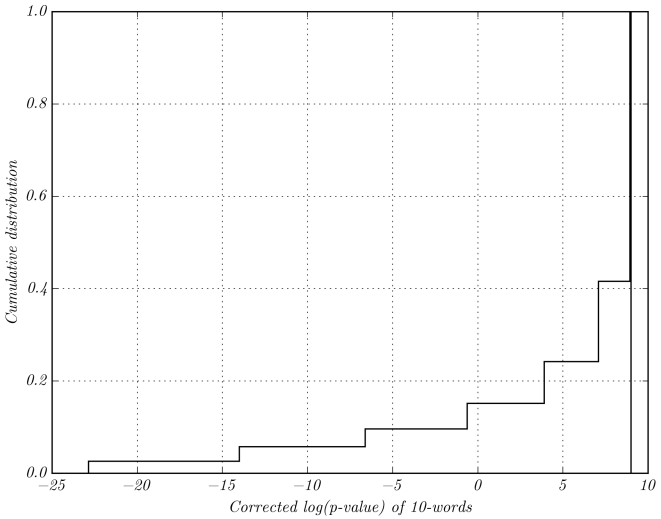
Let w be any k-mer and let L be the sum of the lengths of all reads. Let X_w be the random variable corresponding to the number of occurrences of w. We make the following simplifying assumptions:

1. Ignore read boundaries and assume X_w corresponds to the number of occurrences of w in a random sequence of length L. Further, since $k \ll L$ we can approximate the total number of words by L (instead of $L - k + 1$).
2. Assume any given word w has the same probability of appearing at any position along L. [15] We can thus denote by p the probability that a k-mer from an arbitrary position of the genome is w, for a fixed k and any k-mer w.
3. Assume subsequent nucleotides in any word are independent of one another and all appear uniformly. That is, $p = |\Sigma|^{-k}$.

Given the above, we get the binomial distribution of $X_w \sim B(L, p_w)$. Since L is quite large (at least 10^8 nucleotides) we can approximate the binomial distribution by a normal distribution [16] : $X_w \sim \mathcal{N}\left(Lp_w, \sqrt{Lp_w(1-p_w)}\right)$. A Bonferroni correction of N is also applied where N, the total number of words, is the number of simultaneous hypotheses considered. A significance (log of corrected p-value) threshold of -5 excludes about 10% of all words in the Leishmania data set.

[15] This ignores local interactions. For example if the word at position 1 is `AAAC` then the word at position 2 cannot be `ACAC`.

[16] The limiting process is not Poisson since $B(n, p)$ only converges to a Poisson process as $n \to \infty$ if np is kept constant which is not our case. The normal approximation only demands $n \to \infty$.



III Chaining seeds

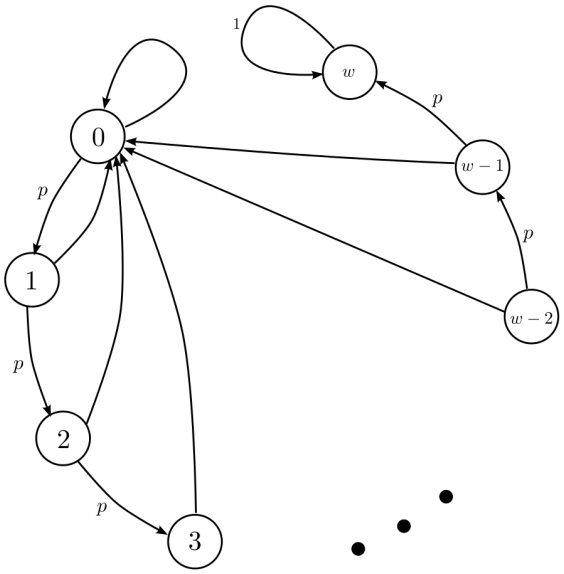
We now consider the following problem: given a set of seeds within a diagonal strip of the dynamic programming table, which partial overlap alignment maximizes the likelihood of the observed seeds, where a partial overlap alignment is a sequence of chainable seeds. This can be reduced (detailed analysis will follow in another report) to a heaviest-path problem over the DAG of all seeds in the given diagonal strip where the weight of edges connecting two seeds is obtained from calculating the the probability that a sub-alignment of length n does not contain any seeds of length k and. We noted earlier that this corresponds to the probability distribution of waiting times between runs of k successes in a sequence of i.i.d Bernoulli trials (where k is the word length). Some combinatorial properties of this distribution, known as the *k-th order geometric distribution*, are known [12]–[15], but they are not computationally useful. We here propose an exact algorithm to calculate this distribution efficiently.

Computational Requirements

We seek the probability distribution $f(n; w)$ of a sub-alignment of length n containing no seed of length w. However, as opposed to the diagonal distance case, we do not have the complication of dependence on diagonal position. Therefore, it suffices to solve a recurrence relation once and use the results for all sequence comparisons.

Probabilistic Model

Consider the following Markov chain where w is the word length and p is the probability of an exact match at any given position of the alignment:



Let $u(n,k)$ be the probability of being at state n at time k. Since $u(n,w)$ is the probability of observing a seed by the n-th step of the alignment the desired distribution $f(n)$ is given by:

$$f(n;w) = 1 - u(n,w)$$

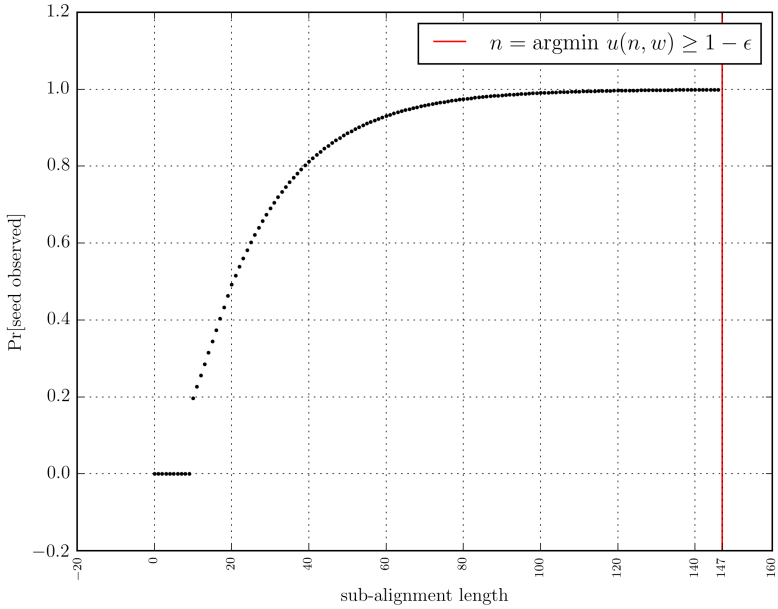
The recurrence relation of $u(n,k)$ for $0 < n < w$ is:

$$u(n,k) = pu(n-1,k-1)$$

with boundary and initial conditions:

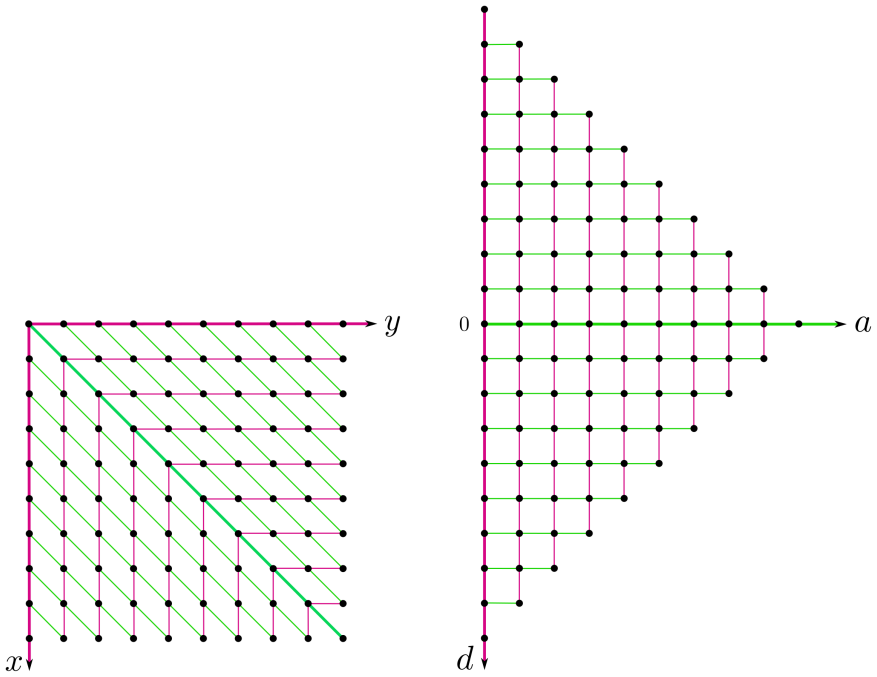
$$u(w,k) = pu(w-1,k-1) + u(w,k-1)$$
$$u(0,k) = 1 - \sum_{n=1}^{w} p(n,k)$$

We know that $f(n;w)$ is decreasing and thus, in practice, we only need to calculate $f(n)$ upto some threshold, say for all n such that $f(n;w) > \epsilon$. We then solve the recurrence relation in increasing order of n and decreasing order of k as long as $u(n,w) < 1-\epsilon$. The following is a plot of $u(n,w)$ for $w = 10$, $p = 0.85$, and $\epsilon = 10^{-3}$. Since the very same distribution is used for all sequence comparisons the computational cost is effectively zero.



IV Mapping Bands to Rectangular Grids

In order to only allocate the necessary memory for the banded overlap alignment we need to map bands (diagonal strips) to rectangular grids in memory. This requires a change of coordinates which maps parallelograms or trapezoids bound by the edges of the dynamic programming table to (roughly) rectangular regions. Two alternatives were previously discussed. Here, we present a refined version of the more convenient of the two: coordinates based on shift and distance from diagonal start cell:



Let (x, y) denote coordinates in the dynamic programming table and (d, a) denote the new coordinates where d is the shift $x - y$ and a is the distance along the starting cell of the d-diagonal. The change of coordinates mapping is given by $\phi : \mathbb{Z}^2 \to \mathbb{Z}^2$:

$$(x, y) \xrightarrow{\phi} \big(x - y, \min(x, y)\big)$$
$$\big(a + \max(d, 0), a - \min(d, 0)\big) \xleftarrow{\phi^{-1}} (d, a)$$

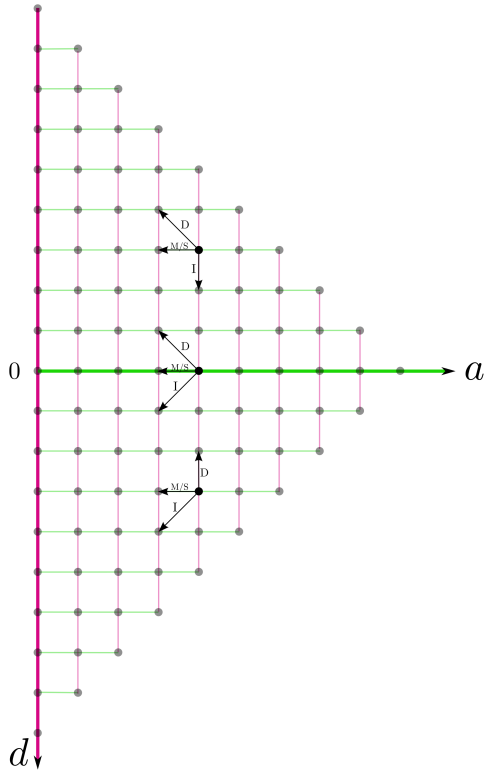
Furthermore, the length of the row at height d in the transformed coordinates is:

$$L(d) = \min(|S| - d, |T|) + \min(d, 0) + 1$$

The alignment band $\mathbf{\Omega}$ is the subset of the grid that needs to be populated. We have:

$$\mathbf{\Omega}_{xy} = \{(x, y); d_{\min} \leq x - y \leq d_{\max}\}$$
$$\mathbf{\Omega}_{da} = \{(d, a); d_{\min} \leq d \leq d_{\max}\}$$

Dynamic programming dependence rules depend on the sign of d:



For simplicity, we populate the dynamic programming table in the natural order of (x, y)-coordinates (while memory is mapped in (d, a)-system). To avoid sweeping the entire (x, y)-grid for in-band cells we use the following bounds:

$$\begin{aligned} \forall (x,y) \in \Omega : \max(0, d_{\min}) &\leq x \leq \min(|S|, |T| + d_{\max}) \\ \forall (x,y) \in \Omega : \max(0, x - d_{\max}) &\leq y \leq \min(|T|, x - d_{\min}) \end{aligned}$$

References

[1] S. Koren, M. C. Schatz, B. P. Walenz, J. Martin, J. T. Howard, G. Ganapathy, Z. Wang, D. A. Rasko, W. R. McCombie, E. D. Jarvis, and A. M. Phillippy, "Hybrid error correction and de novo assembly of single-molecule sequencing reads," *Nat Biotech*, vol. 30, no. 7, pp. 693–700, 2012. DOI: 10.1038/nbt.2280.

[2] C.-S. Chin, D. H. Alexander, P. Marks, A. A. Klammer, J. Drake, C. Heiner, A. Clum, A. Copeland, J. Huddleston, E. E. Eichler, S. W. Turner, and J. Korlach, "Nonhybrid, finished microbial genome assemblies from long-read smrt sequencing data," *Nat Meth*, vol. 10, no. 6, pp. 563–569, 2013, Article.

[3] K. Berlin, S. Koren, C.-S. Chin, J. P. Drake, J. M. Landolin, and A. M. Phillippy, "Assembling large genomes with single-molecule sequencing and locality-sensitive hashing," *Nat Biotech*, vol. 33, no. 6, pp. 623–630, 2015, Research.

[4] C. Ye and S. Ma, "Sparc: A sparsity-based consensus algorithm for long erroneous sequencing reads," PeerJ PrePrints, Tech. Rep., 2015.

[5] H. Li and R. Durbin, "Fast and accurate short read alignment with burrows–wheeler transform," *Bioinformatics*, vol. 25, no. 14, pp. 1754–1760, 2009.

[6] M. J. Chaisson and G. Tesler, "Mapping single molecule sequencing reads using basic local alignment with successive refinement (blasr): Application and theory," *BMC bioinformatics*, vol. 13, no. 1, p. 238, 2012.

[7] C. Notredame, D. G. Higgins, and J. Heringa, "T-coffee: A novel method for fast and accurate multiple sequence alignment," *Journal of molecular biology*, vol. 302, no. 1, pp. 205–217, 2000.

[8] C. Lee, C. Grasso, and M. F. Sharlow, "Multiple sequence alignment using partial order graphs," *Bioinformatics*, vol. 18, no. 3, pp. 452–464, 2002.

[9] C. Grasso and C. Lee, "Combining partial order alignment and progressive multiple sequence alignment increases alignment speed and scalability to very large alignment problems," *Bioinformatics*, vol. 20, no. 10, pp. 1546–1556, 2004.

[10] M. Brudno, M. Chapman, B. Göttgens, S. Batzoglou, and B. Morgenstern, "Fast and sensitive multiple alignment of large genomic sequences," *BMC bioinformatics*, vol. 4, no. 1, p. 66, 2003.

[11] L. Noé and G. Kucherov, "Improved hit criteria for dna local alignment," *BMC bioinformatics*, vol. 5, no. 1, p. 149, 2004.

[12] W. Feller, *An introduction to probability theory and its applications*, ser. Wiley series in probability and mathematical statistics: Probability and mathematical statistics. Wiley, 1971.

[13] S. Aki, H. Kuboki, and K. Hirano, "On discrete distributions of order k," *Annals of the Institute of Statistical Mathematics*, vol. 36, no. 1, pp. 431–440, 1984.

[14] A. N. Philippou, C. Georghiou, and G. N. Philippou, "A generalized geometric distribution and some of its properties," *Statistics & Probability Letters*, vol. 1, no. 4, pp. 171–175, 1983.

[15] M. J. Barry and A. J. L. Bello, "The moment generating function of the geometric distribution of order k," *The Fibonacci Quarterly*, vol. 31, pp. 178–180, 1993.