Seqpare: a self-consistent metric of similarity between genomic interval sets

Selena C. Feng[1], Nathan C. Sheffield[3], and Jianglin Feng[2, 3,*]

[1] Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology

[2] Deepstanding LLC, Crozet, Virginia

[3] Center for Public Health Genomics, University of Virginia

[*] To whom correspondence should be addressed.

ABSTRACT

Summary: Searching genomic interval sets produced by sequencing methods has been widely and routinely performed; however, existing metrics for quantifying similarities among interval sets are inconsistent. Here we introduce *Seqpare*, a self-consistent and effective metric of similarity and tool for comparing sequences based on their interval sets. With this metric, the similarity of two interval sets is quantified by a single index, the ratio of their effective overlap over the union: an index of *zero* indicates unrelated interval sets, and an index of *one* means that the interval sets are identical. Analysis and tests confirm the effectiveness and self-consistency of the *Seqpare* metric.

Availability: https://github.com/deepstanding/seqpare

Contact: jf.xtable@gmail.com

INTRODUCTION

Functional genomic data are often summarized as interval sets and deposited in public repositories (e.g., UCSC, ENCODE, Roadmap, GEO, SRA etc.). Identifying relationships among sequences and searching through widely available sequence data are routine tasks in genomic research. A fundamental operation in genomic/epigenomic analysis is comparing two interval sets, and many algorithms and tools have been developed for this purpose (Alekseyenko & Lee, 2007; Kent, 2002; Quinlan & Hall 2010; Li, 2011; Neph etal, 2012; Cormen et al. 2001; Jalili et al., 2019; Feng et al., 2019). These methods are based on computing the total number of intersections (overlaps) between the two interval sets. To compare a query interval set with multiple interval sets in a genomic sequence database, searching tools LOLA (Sheffield & Bock, 2016) and GIGGLE (Layer et al., 2018) calculate two values — Fisher's exact *p*-value and the odds-ratio based on the total number of intersections — and use them as the similarity score to rank the search results. These similarity metrics have proven useful for determining relationships among interval sets, but also have some flaws. First, calculating the Fisher's exact test results requires building a contingency table, but determining its values is not straightforward. The *p*-value and odds-ratio for two intervals sets (with number of intervals N_1 and N_2) are calculated from four numbers, namely, the number of intersections between the two sets, n, the number of intervals in set 1 that do not overlap an interval in set 2, $N_1 - n$, the number of intervals in set 2 that do not overlap an interval in set 1, $N_2 - n$, and the number of intervals that are not present in either set, m. Determining the value of the fourth number m is not straightforward; in LOLA, it depends on the definition of a "universe set" that is not objectively defined, whereas GIGGLE estimates m from the two interval sets. Second, the total number of overlaps n does not necessarily reflect similarity since intervals can have very different lengths (often in the range of $1 \sim 10^5$ base pairs) and two very different intervals may intersect by only a few base pairs. This can result in inconsistency of the metrics: a comparison between two identical interval sets may have a larger *p*-value or smaller odds-ratio than a

comparison between two different interval sets (see example cases and analysis in the next section). More strikingly, since one interval may contain or cover other intervals in an interval set, depending on how the overlaps are computed, n can be larger than N_1 and/or N_2, i.e., N_1-n and/or N_2-n can be negative, which leads to both the p-value and odds-ratio being undefined—another potential source of inconsistency. Third, the Fisher's exact-based metrics require two values (p-value and odds-ratio) but neither is a *direct* measurement of the similarity: p-values are sensitive to the total number of regions and can range as low as 10^{-200} for large genomic interval sets, and odds-ratios are sensitive to small numbers; and neither metric directly informs on *how similar* the two sets are. Last, the p-value calculation is computationally expensive for genomic interval sets, particularly when the number of intervals is large (up to 10^9). To overcome these weaknesses of the Fisher's exact-based metrics, we developed *Seqpare*, a self-consistent metric for quantifying the similarity among genomic interval sets.

METHOD

Seqpare metric

The *Seqpare* metric uses a single index to quantify the degree of similarity S of two interval sets with number of intervals N_1 and N_2. Similar to the *Jaccard* index, the S*eqpare* metric is directly defined as the ratio of the total effective overlap O of the two interval sets over the union $N_1+N_2 - O$:

$$S = O / (N_1 + N_2 - O) \qquad (1)$$

For two intervals v_1 in set 1 and v_2 in set 2, the similarity s is defined as:

$$s = o / (l_1 + l_2 - o) \qquad (2)$$

where o is the length of the intersection and l_1 and l_2 are the lengths of v_1 and v_2 respectively. Definition 2 is the *Jaccard* index for individual intervals: o represents the effective overlap of the two intervals and s takes values in the range of [0, 1]: s = 0 indicates that there is no overlap between the two intervals, and s = 1 means that the union equals the overlap so v_1 and v_2 are identical. Then the total

effective overlap O for the two interval sets can be calculated by adding up the similarities of all mutual best matching (MBM) pairs:

$$O = \sum s_{MBM} \qquad (3)$$

A MBM pair is defined as a pair of intervals v_1 and v_2 that fulfill the following conditions: among all intervals in set 2 that intersect v_1, v_2 matches v_1 the best, i.e., the similarity s between v_1 and v_2 is the highest among those intersections; and among all intervals in set 1 that intersect v_2, v_1 matches v_2 the best. Clearly, if two intervals only intersect each other, then they form an MBM pair. In Figure 1a, the two long intervals (the 1st in set 1 and the 1st in set 2) only intersect each other (intersection pair $ip_{1,1}$), so they form an MBM pair; similarly the two short intervals $ip_{2,2}$ form another MBM pair. For intervals that involve multiple intersections, we define a relatively simple and strict rule to find the MBM pairs: find and choose the first MBM pair that has the highest s among all involved intersection pairs, then find and choose the next MBM pair that has the highest s from the rest of the intersection pairs (excluding all pairs that involve the intervals that are already chosen), and so on until there are no more intersection pairs. In Figure 1b, there are 3 intersection pairs: $ip_{1,2}$ with s=1/5, $ip_{1,3}$ with s=1/9, and $ip_{2,3}$ with s=1/5. So the first MBM pair is either $ip_{1,2}$ or $ip_{2,3}$ depending on which one is found first. If $ip_{1,2}$ is chosen as the first MBM pair, then $ip_{1,3}$ will not be considered since interval 1 in set 1 is already chosen, and then we have only $ip_{2,3}$ left, which is the second MBM pair. We get the same result if $ip_{2,3}$ is chosen as the first MBM pair. In Figure 1d, there are 6 *ip*s and two *MBM pairs:* $ip_{1,1}$ *and* $ip_{3,2}$ both with s=1, where interval 2 in set 1 ($i_{2,1}$) does not have a match. Note that interval $i_{2,1}$ matches best with interval $i_{1,2}$, but $i_{1,2}$ does not match best with $i_{2,1}$, so they are not the *mutual* best matching pair.

Since the number of total matching pairs ≤ Min(N_1, N_2) — the minimum of N_1 and N_2 — and s is in range of [0, 1], we obtain $O \leq$ Min(N_1, N_2), and S takes a value in the range of [0, 1]. If S is zero, then there is no matching pair, and vice versa; if S = 1, then $N_1 = N_2 = O$ (the two sets are equivalent), and vice versa. And, because each s is the *mutual* best match, O is symmetric (the amount of overlap between set 1 and set 2 is the same as that between set 2 to set 1) and so is S.

In Figure 1a, O_a=2 and S_a=1, which is correct because the two sets are identical. In Figure 1b, O_b=2/5 and S_b=1/14, which is expected since the two sets are very different. The Fisher's exact approach is inconsistent here: the *p*-value in 1*b* is smaller than that in 1*a* although the two sets in 1*b* are very different while those in 1*a* are equivalent. Assuming that the number of intervals *N* in the 'universe set' is 100, then Fisher's exact test contingency table is [(2, 0), (0, 98)] in 1*a* and [(3, 0), (0, 97)] in 1*b*, which gives $p_a = 2.02\text{x}10^{-4}$ and $p_b = 6.18\text{x}10^{-6}$ respectively. The odds-ratio is ∞ in both cases. In cases *c* and *d*, N_1 - *n* and N_2 - *n* are all negative, so it is not conceptually appropriate to use Fisher's exact test to calculate the *p*-value and odds-ratio.

Implementation

The implementation of the *Seqpare* metric is simple. The searching for MBM pairs is deterministic and it can be implemented by directly following the description in the above section. The *Seqpare* code is built on top of the AIList (Feng et al., 2019) software written in C.

RESULT

A test with real genomic interval sets

To test *Seqpare* and compare it with the Fisher's exact-based metrics, we took 100 interval sets from a UCSC database and used one interval set, *affyGnf1h*, as a query to search over the database. Because the database contains the query set, *affyGnf1h* should have the highest similarity score. Table 1 shows part of the result. Interval set *affyGnf1h* indeed ranks first with maximum similarity 1 when using *Seqpare*, but it ranks 94th out of 100 when using the *p*-value and ranks last when using the odds-ratio. This happens because N_1-*n* and N_2-*n* are both negative (n=16686, N_1=N_2=12158). Given this inconsistency, GIGGLE sets negative N_1-n and N_2-*n* to zero to calculate the *p*-value, and to one to calculate the odds-ratio. The *Seqpar*e indices for other interval sets are all small (<0.03) because

the average effective overlap of an intersection pair in those sets is about 0.1 or less, i.e., they are very different from the query set affyGnf1h; however, all of the *p*-values are so small (e^{-200}), which suggests that the *p*-value is not a meaningful similarity index for these genomic interval sets. This search takes 6m30s for *Seqpare* and 15m32s for GIGGLE. All computations were carried out on a computer with a 2.8GHz CPU, 16GB memory, and an external SSD hard disk. The complete results can be found at the same site as the software.

CONCLUSION

We have shown that the Fisher's exact test approach may be not the most appropriate test statistic for comparing similarity among interval sets. While the approach has been shown to be successful for many questions, we have demonstrated how it can break down for a variety of reasons, such as very similar interval sets, within-set containment, widely varying interval lengths among sets, or small effective overlaps. In contrast, *Seqpare* is a self-consistent metric for quantifying the similarity of two interval sets that addresses these concerns. *Seqpare* is the first rigorously defined metric for comparing two sequences based on their interval sets. In addition to the metric itself, our *Seqpare* software tool provides functions for both searching and mapping large-scale interval datasets. We anticipate that this approach will contribute to novel results for interval set searching.

References

Alekseyenko, A.V. and Lee, C.J. (2007) Nested containment list (NCList): A new algorithm for accelerating interval query of genome alignment and interval databases. *Bioinformatics*, **23**, 1386–1393.

Cormen, T.H. et al. (2001) Introduction to algorithms second edition.

Feng, J. et al. (2019) Augmented interval list: A novel data structure for efficient genomic interval search. *Bioinfomatics*, **35**, 4907-4911

Giardine, B. et al. (2005) Galaxy: A platform for interactive large-scale genome analysis. *Genome Res.*, **15**, 1451–1455.

Jalili,V. et al. (2019) Next generation indexing for genomic intervals. *IEEE Transactions on Knowledge and Data Engineering*, **31**, 2008–2021.

Layer, R.M. et al. (2018) GIGGLE: A search engine for large-scale integrated genome analysis. *Nature Methods*, **15**, 123–126.

Li, H. Tabix: fast retrieval of sequence features from generic TAB-delimited files. *Bioinformatics,* **27**, 718-719. 2011.

Kent, W.J. et al. (2002) The human genome browser at ucsc. *Genome Res.*, **12**, 996–1006.

Neph, S. et al. (2012) BEDOPS: high-performance genomic feature operations. *Bioinformatics*, **28**, 1919–1920.

Quinlan, A.R. and Hall, I.M. (2010) BEDTools: A flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**, 841–842.

Richardson, J.E. (2006) Fjoin: Simple and efficient computation of feature overlaps. *Journal of Computational Biology*, **13**, 1457–1464.

Sheffield, N.C. and Bock, C. (2016) LOLA: Enrichment analysis for genomic region sets and regulatory elements in R and bioconductor. *Bioinformatics*, **32**, 587–589.

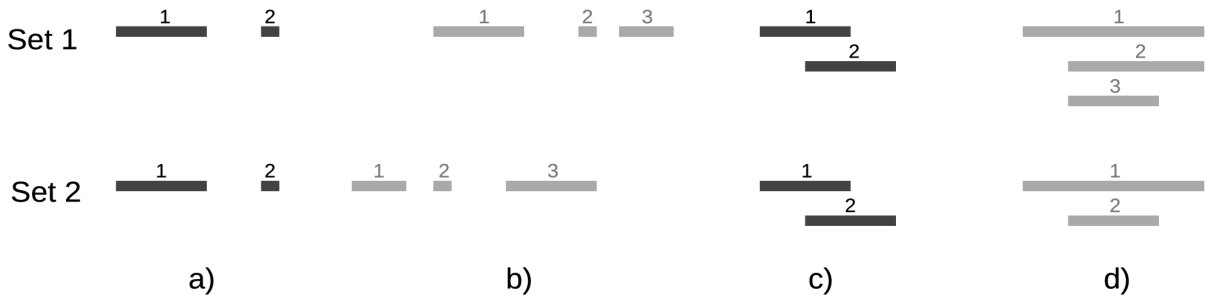

Figure 1. Example cases for illustrating the Seqpare similarity metric. The length ratio of the short interval to the longer intervals are 1: 5 in a), 1: 3: 5 in b), and 2: 3 :4 in d). The total number of overlaps n is 2 for a), 3 for b) (interval 1 in Set 1 intersects two intervals in Set 2), 4 for c) and 6 for d). The p-value in case b is smaller than that in case a. N_1 *- n and* N_2 *- n are both negative in cases c and d.*

***Seqpare* metric**		***p*-value and odds-ratio**				**Interval dataset file name**
rank	**similarity**	**rank_p**	***p*-value**	**rank_or**	**odds-ratio**	
1	1.0	94	$1.543\ e^{-197}$	100	$9.046\ e^{-16}$	affyGnf1h
2	0.029	19	$8.328\ e^{-201}$	1	25.25	ccdsGene
3	0.026	35	$2.743\ e^{-200}$	3	4.425	allenBrainAli
4	0.022	39	$3.372\ e^{-200}$	32	$7.972\ e^{-11}$	affyU133Plus2
5	0.021	4	$5.403\ e^{-202}$	2	17.24	affyU133

Table 1. Comparison of Seqpare and GIGGLE similarity metrics: partial list of the search results from a collection of 100 interval sets, which contains the query set affyGnf1h.