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

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

[1] Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology
[2] Deepstanding LLC, Crozet, Virginia

[*] To whom correspondence should be addressed.

ABSTRACT

Summary: Searching genomic interval sets produced by sequencing methods has been widely and routinely performed; however, a rigorous metric for quantifying similarities among interval sets is still lacking. 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, which directly represents the percentage of their effective overlap: an index of *zero* indicates totally unrelated interval sets, and an index of *one* means that the interval sets are equivalent. 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

Genomic sequences are usually summarized and deposited as interval sets in public repositories (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 some critical flaws for genomic interval sets. First, 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 non-overlapping intervals in set 1, $N_1 - n$, the number of non-overlapping intervals in set 2, $N_2 - n$, and the number of intervals that are not present in either set, m. Determining the value of the fourth number m is not straightforward and it depends on the definition of the 'universe set' (in LOLA the universe set is a user-specified interval set, where GIGGLE directly calculates m from the two involved interval sets), and the errors of the *p*-value and odds-ratio calculated from such an estimated m are not evaluable. Second, the total number of overlaps n does not reflect the number of similar intervals 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: two equivalent sets may have a larger/equal *p*-value or smaller/equal odds-ratio than two different sets do (see example cases and analysis in the next section). More strikingly, since one

interval may contain or cover other intervals in an interval set, n can be larger than N_1 and/or N_2, i.e., n-N_1 and/or n-N_2 can be negative, which leads to both the p-value and odds-ratio being undefined—another case of inconsistency of the GIGGLE p-metric (this does not apply to LOLA p-metric because of the use of a user-defined universe set). Third, the p-based metrics make use two values (p-value and odds-ratio) but neither is a *direct* measurement of the similarity: a given p-value (usually in the range of 10^{-20} to 10^{-200} for genomic interval sets) and/or odds-ratio do *not tell us how similar* the two sets are. Last, the p-value calculation is computationally expensive for genomic interval sets since the involved numbers are usually very big (up to 10^9). To overcome these flaws of the p-based metrics, we developed *Seqpare*, a self-consistent metric for quantifying the similarity among genomic interval sets.

RESULTS

Seqpare metric and implementation

The *Seqpare* metric uses a single index to quantify the degree of similarity S of two interval sets. 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 $N_1+N_2 - O$:

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

Where O fulfills the following two requirements:

- *the resulting S can only take a value in the range of [0, 1]: S=0 if and only if the two sets do not have any overlap; S=1 if and only if the two set are exactly the same (two equivalent sets have the maximum similarity).*
- *O itself is symmetric, i.e., the amount of overlap between set 1 and set 2 is the same as that between set 2 and set 1. This makes S symmetric.*

We first consider the number of intersections n as O. The two interval sets in Figure 1a are equivalent with 2 intersections and S_a *is* 1 according to equation 1. In Figure 1b the two sets are so different that

the similarity index should be much smaller than 1 if the metric is self-consistent, but there are 3 intersections and the union is 3, so S_b is also 1. More dramatically, the index $S_c = \infty$ in case *c* (the two sets are equivalent but the union is 0), and $S_d = -6$ in case *d* (the two sets are similar but the union is -1). Therefore, the similarity metric based on the total number of intersections *n* is not self-consistent. The

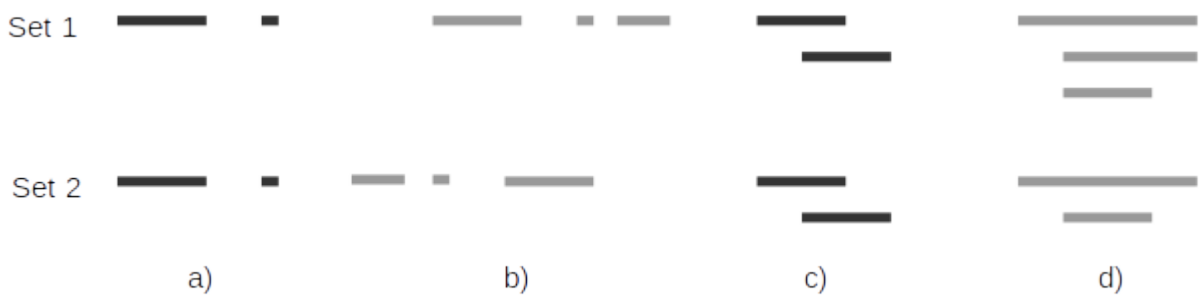

Figure 1. Example cases for illustrating the Seqpare similarity metric. The length ratio of the short interval to the longer intervals are 1: 5 in a), 1: 3: 5 in b), and 2: 3 :4 in d). The total number of overlaps n is 2 for a), 3 for b) (the longer interval in Set 1 intersects two intervals in Set 2), 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.

p-value in case *b* is smaller than that in case *a* although the two sets in *b* are very different while those in *a* are equivalent: assume 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 case *a* and [(3, 0), (0, 97)] in case *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 even conceptually appropriate to use Fisher's exact test to calculate the *p*-value and odds-ratio. To overcome this inconsistency, we have to consider the similarity at the individual interval level. We define the similarity *s* of two intervals v_1 in set 1 and v_2 in set 2 as:

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

where *o* is the length of the overlap 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 they are equivalent.

Now, the critical step is to find the connection between O and s. A simple way is to calculate the s values for all of the intersections and add them up to obtain O. This gives O_b = 1/5+1/9+1/5 and S_b ~ 0.1, which is smaller than S_a = 1. However, in Figure 1c each interval in set 1 intersects two intervals in set 2, and each interval in set 2 also intersects two intervals in set 1. This is a many-to-many intersection case, where O = 8/3 (=1+1/3+1/3+1) and S = 2 >1, which does not satisfy the above requirement 1. To correct this we make the following final rules: for intervals that involve multiple intersections, find and choose the *mutual* best matching pairs (with the highest s), and then find the next *mutual* best matching pairs from the *rest* of the intervals (where *any interval can only be matched once*), and so on until there are no more matching pairs; the overall effective overlap O is, then, the sum of the s values of all of the matching pairs:

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

In Figure 1c, there are two perfect matching pairs, so O = 2 and S_c = 1 as expected. Now, if we reconsider the case in Figure 1b, there are two mutual best matching pairs and each has s = 1/5, so O = 2/5 and S_b ~ 0.07, which indicates that the two interval sets are indeed very different. In Figure 1d, the second interval in set 1 matches the best (s = 3/4) with the first interval in set 2, but they are not the *mutual* best match since the first interval of set 2 is a perfect and mutual best match for the first interval of set 1; because the third interval in set 1 is a mutual best match to the second interval in set 2, the second interval in set 1 will not have a match, so we have O = 2 and S = 2/3.

In general, 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 $N_1 = N_2 = O$, and vice versa. Thus, requirement 1 for O is satisfied. And, because each s is the *mutual* best match, O is symmetric (as well as deterministic), so requirement 2 is satisfied.

The implementation of the *Seqpare* metric is simple, except for the searching for the mutual matching pairs in the case of many-to-many intersections, which can be implemented as a recursive

function. The *Seqpare* code is built on top of the AIList (Feng et al., 2019) software written in C.

A test with real genomic interval sets

To test *Seqpare* and compare it with the *p*-value/odds-ratio metric of GIGGLE, we take 100 interval sets from the UCSC data source as a database and use one of the interval sets *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-N_1 and n-N_2 are both negative

***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.

(n=16686, N_1=N_2=12158 from GIGGLE; instead of outputting an error message in this case, GIGGLE forces negative n-N_1 and n-N_2 to zero to calculate the *p*-value, and changes them to 1 to calculate the odds-ratio). The *Seqpar*e indices f or other interval sets are all small (<0.03) because the average effective overlap for an intersection 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

Seqpare is a self-consistent metric and tool for quantifying the similarity of two interval sets, and the software tool provides functions for searching large-scale interval datasets. *Seqpare* is the first *rigorously* defined metric for comparing two interval sets. Analysis and tests confirm the effectiveness and self-consistency of the *Seqpare* metric.

ACKNOWLEDGMENT

The author JF would like to thank Dr Nathan Sheffield of the University of Virginia for some useful discussion.

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.