Mobile DNA

SOFTWARE

Open Access

REPrise: de novo interspersed repeat detection using inexact seeding

Atsushi Takeda[1,2†], Daisuke Nonaka[3†], Yuta Imazu[4], Tsukasa Fukunaga[3,5*] and Michiaki Hamada[1,2,6*]

Abstract

Background Interspersed repeats occupy a large part of many eukaryotic genomes, and thus their accurate annotation is essential for various genome analyses. Database-free de novo repeat detection approaches are powerful for annotating genomes that lack well-curated repeat databases. However, existing tools do not yet have sufficient repeat detection performance.

Results In this study, we developed REPrise, a de novo interspersed repeat detection software program based on a seed-and-extension method. Although the algorithm of REPrise is similar to that of RepeatScout, which is currently the de facto standard tool, we incorporated three unique techniques into REPrise: inexact seeding, affine gap scoring and loose masking. Analyses of rice and simulation genome datasets showed that REPrise outperformed RepeatScout in terms of sensitivity, especially when the repeat sequences contained many mutations. Furthermore, when applied to the complete human genome dataset T2T-CHM13, REPrise demonstrated the potential to detect novel repeat sequence families.

Conclusion REPrise can detect interspersed repeats with high sensitivity even in long genomes. Our software enhances repeat annotation in diverse genomic studies, contributing to a deeper understanding of genomic structures.

Keywords De novo repeat detection, Seed-and-extend, Inexact seed, REPrise

Introduction

Interspersed repeats, which are mainly amplified by copying of transposable elements (TEs) while undergoing evolutionary mutations, constitute a significant portion of many eukaryotic genomes. For example, they account for 54% of the human genome [1] and 85% of the wheat genome [2]. These repeat sequences were once considered functionless 'junk' regions, but they are now understood to play various roles in cellular processes, including RNA processing and transcriptional regulation [3–5]. Moreover, because these repeat sequences contain phylogenetic signals, they are employed as markers for reconstructing species trees [6]. Consequently, the accurate annotation of interspersed repeats is an essential task of genome analysis.

There are three main computational approaches for identifying interspersed repeats: database-dependent,

†Atsushi Takeda and Daisuke Nonaka contributed equally to this work.

*Correspondence:
Tsukasa Fukunaga
fukunaga@aoni.waseda.jp
Michiaki Hamada
mhamada@waseda.jp
[1] Department of Electrical Engineering and Bioscience, Graduate School of Advanced Science and Engineering, Waseda University, Tokyo 1698555, Japan
[2] Computational Bio Big-Data Open Innovation Laboratory, AIST-Waseda University, Tokyo 1698555, Japan
[3] Department of Computer Science, Graduate School of Information Science and Technology, the University of Tokyo, Tokyo 1130032, Japan
[4] Department of Electrical Engineering and Bioscience, School of Advanced Science and Engineering, Waseda University, Tokyo 1698555, Japan
[5] Waseda Institute for Advanced Study, Waseda University, Tokyo 1690051, Japan
[6] Graduate School of Medicine, Nippon Medical School, Tokyo 1138602, Japan

signature-based and de novo database-free methods [7]. Database-dependent methods annotate the repeats by aligning sequences from repeat databases, such as Repbase [8] and Dfam [9], to the genome. The representative tool for this approach is RepeatMasker [10], which is widely used for annotating repeats in newly sequenced genomes [1]. However, this approach inherently faces challenges in annotating genomes without well-curated repeat databases and in detecting repeats absent from existing databases. Signature-based methods search repeat sequences for specific classes based on particular sequence motifs and structures. The representative tools are LTR_FINDER [11], which detects Long Terminal Repeat (LTR) retrotransposons based on the specific repeat structure, and TSDFINDER [12], which targets target site duplication sequences of non-LTR retrotransposons. These methods can detect novel interspersed repeats of classes that possess the target structures. However, they cannot detect interspersed repeats that do not have the target structures or whose structure has not been conserved due to the mutation, fragmentation, or nesting.

Recent advances in sequencing technology have led to numerous cases that cannot be adequately handled by this database-dependent approach alone. For example, the Earth BioGenome Project [13], which aims to sequence all eukaryotic genomes by 2030, continues to sequence the genomes of many non-model organisms that lack sufficient repeat annotation. As another example, the Telomere-to-Telomere (T2T) Consortium has released a complete human genome using long read sequencers and has discovered novel repeat families in newly sequenced regions [1, 14].

In these cases, de novo database-free methods are practical for detecting interspersed repeats. At present, various tools based on this approach have been developed [15–20], and benchmark studies have indicated that RepeatScout [21] is the best-performing tool [22–24]. RepeatScout employs a seed-and-extension method, originally proposed as a fast alignment heuristic in BLAST [25]. RepeatScout first identifies frequently occurring seed regions within the genome and then performs extension alignments from both ends of these seed regions. Once the extension alignment is completed, the corresponding seed regions are masked, and the extension alignment is performed again from different seed regions. This cycle of alignment and masking continues until alignments from all seed regions have been performed. Because of its high repeat detection performance, RepeatScout has been incorporated into comprehensive repeat annotation pipelines such as EDTA [23] and RepeatModeler2 [26]. However, even RepeatScout has yet to achieve perfect repeat detection, indicating that there is room for further improvement in the algorithm. In addition, due to implementation issues, RepeatScout cannot be applied directly to long genome sequences such as the human genome.

In this study, we developed REPrise (REPeat Recognition using Inexact Seed-and-Extension), a tool that identifies interspersed repeats de novo with higher sensitivity than RepeatScout. The algorithm of REPrise is organized into three main steps similar to those in RepeatScout: seed detection, extension alignment, and masking. REPrise improved each of these steps compared to RepeatScout as follows.

1. Unlike RepeatScout, which used exact k-mer matches as seed sequences, REPrise employed inexact seeds that allow for d mismatches. Because some highly sensitive sequence alignment tools utilized seeding that is not exact matches [27], we expected that the inexact seeding improves the detection sensitivity of interspersed repeats. Note that phRAIDER [19] used seeding that is not exact matches for interspersed repeat detection, but phRAIDER is limited in the detectable repeat types.
2. For the indel score in the extension alignment, RepeatScout employed a linear gap penalty, whereas REPrise used an affine gap penalty [28], which is a more commonly used scoring system for sequence alignments.
3. In the masking step, RepeatScout masked all seeds within the identified repeats, whereas REPrise only masked the seed used for repeat detection. In other words, REPrise reduces the number of regions being masked, preserving more candidate regions for subsequent repeat detection.

In addition, we implemented REPrise to be applicable to long genome sequences. Our validation experiments showed that REPrise outperformed RepeatScout in the detection sensitivity of interspersed repeats using both rice and simulation genome datasets. We also applied REPrise to the complete human genome T2T-CHM13 and identified novel TE family candidates that have not yet been annotated.

Methods

Overview of the REPrise algorithm

Both REPrise and RepeatScout take genome sequences as input and output consensus sequences for each identified repeat families. The consensus sequence is a string composed of four characters A, T, C, G. Figure 1 provides an overview of the REPrise and RepeatScout algorithms. Both algorithms first construct a seed table composed of seed sequences with their frequencies in the input

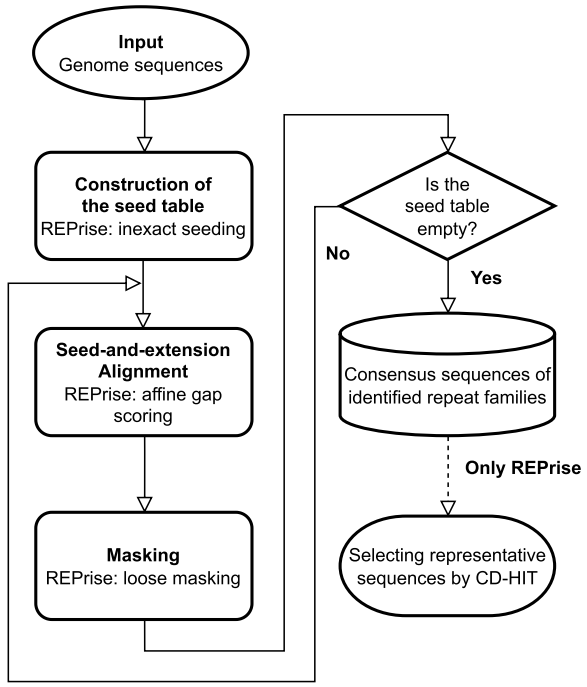

Fig. 1 The schematic illustration of REPrise and RepeatScout algorithms. These algorithms first construct a seed table from the input genome sequences. REPrise utilizes inexact seeds, frequently appearing *k*-mers permitting *d* substitutions, for the table construction. Subsequently, these algorithms perform seed-and-extension alignments on both ends of the seeds. REPrise adopts the affine gap scoring in this step. These algorithms then mask seed regions in the detected repeat regions. REPrise performs looser seed masking than RepeatScout. This cycle of alignment and masking is repeated until the seed table is depleted. In REPrise, representative sequences are selected from the consensus sequences of repeat families using CD-HIT and the representative sequences are outputted

genome. As the seed sequences, RepeatScout uses *k*-mers that occur more than *c* times in the genome without allowing substitutions. In contrast, REPrise allows for *d* substitutions. In accordance with a previous benchmark study [23], we set the value of *c* to 10. In the next step, the extension alignment is performed on the most frequent seeds in the seed table. Each alignment result from one seed sequence corresponds to one repeat family. While RepeatScout employs linear gap scoring, REPrise uses affine gap scoring. After the alignment, the genome regions corresponding to these seed sequences are masked, and the masked seeds are also removed from the seed table. REPrise adopts a masking approach that only masks the seed used for repeat detection, resulting in fewer masked regions compared to RepeatScout. Then, the selection of the most frequently occurring seed and the extension alignment is performed again. This cycle of the seed-and-extension and masking is repeated until no more seeds are left in the seed table. Finally, REPrise further merges the consensus sequences of the identified repeat families using CD-HIT [29].

Construction of the seed table

While RepeatScout constructs the seed table by scanning the genome only once, REPrise cannot adopt this approach because it employs inexact seeding. In addition, REPrise has to ensure that a *k*-mer is not counted multiple times from different seeds. To address these issues, REPrise uses a suffix array to index the genome sequence *T*. The suffix array is a data structure that lists starting positions of all suffixes of a given string in the lexicographical order. It can be constructed with the time complexity of $O(|T|)$ using the induced sorting algorithm [30] and is frequently used in bioinformatics sequence analysis software [31]. By performing a binary search on the suffix array, for a given *k*-mer, we can count the frequencies of all *d*-similar *k*-mers in the genome. We defined two *k*-mers whose Hamming distance is less than or equal to *d* as *d*-similar *k*-mers.

REPrise constructs the seed table in the following manner:

1. The suffix array of the genome sequence is constructed.
2. For each *k*-mer seed, REPrise counts the frequencies of all *d*-similar *k*-mers in the genome. The count is recorded as the frequency for each *k*-mer seed. In this step, REPrise allows that a *k*-mer is counted multiple times from different seeds. To accelerate this process, we employed parallel computation using multi-threading with OpenMP.
3. The *k*-mer seeds are sorted based on their frequencies.
4. In this sorted order, REPrise recounts the frequencies of all *d*-similar *k*-mers in the genome for each *k*-mer seed. In this recount step, REPrise does not count *k*-mer once counted again and thus can avoid multiple counts of the same *k*-mer from different *k*-mer seeds. Finally, only the seeds with a frequency of *c* or higher are retained in the seed table.

Seed-and-Extension alignment with affine gap scoring

The extension step of REPrise uses the banded alignment with affine gap scoring. The alignment procedure between two sequences is detailed in the Supplementary Material (Section S1). The consensus sequence *Q* among repeats is obtained by extending sequences from the seed region using the banded alignment. As the example, we introduce the rightward extension algorithm. Let Q_t denote the consensus sequence exteneded by *t* nucleotides to the right from the seed region. Given that the

seed sequence is located at M positions in the genome, and the M sequences adjacent to the right of the seed region are defined as $S_1, S_2, \ldots, S_M$. The extension of the consensus sequence from Q_t to Q_{t+1} is performed according to the following equation:

$$Q_{t+1} = Q_t \cdot \underset{N \in \{A,T,G,C\}}{\operatorname{argmax}} A(Q_t \cdot N; S_1, \ldots, S_M),$$

$$A(X; S_1, \ldots, S_M) = \sum_{m=1}^{M} \max \begin{cases} \max_j (a_{X,S_m}(|X|, j)) \\ \max_{1 \le i < |X|, j} (a_{X,S_m}(i,j) + p) \\ 0 \end{cases}$$

where $\cdot$ is an operator that appends a character to the right end of a sequence, $A(X; S_1, \ldots, S_M)$ is summation of the alignment score between X and each S_m, and $a_{X,S_m}(i,j)$ is the alignment score between between the first i characters of X and the first j characters of S_m. The first term of A indicates that X aligns with S_m up to the end of X. The second term of A implys that X aligns only up to the middle of S_m. p is an incomplete-fit penalty, which is set to -20, consistent with the value used in RepeatScout. The third term of A represents that X and S_m are not aligned at all.

The extension of Q_t is halted when t_{max} is not updated over a predefined length w. t_{max} is updated to t when the following conditions are satisfied:

$$A(Q_t; S_1, \ldots, S_M) \ge A(Q_{t_{max}}; S_1, \ldots, S_M) + r(t - t_{max}),$$

where r represents the penalty associated with the number of regions to be extended as the repeat regions in the genome and is set to three as with RepeatScout. When the extention is terminated, $Q_{t_{max}}$ is concatenated to the right side of the seed sequence. In this study, w is set to 100 as with RepeatScout. The same extension process is performed to the left direction, and the final consensus sequence Q is obtained. As the left and right extensions are independent processes, they were computed in parallel by multi-threading with OpenMP.

Loose masking

Masking is a step to remove seed regions once used for repeat detection from the subsequent analysis, avoiding redundancy in repeat detection and reducing the computation time. Specifically, the consensus sequence obtained in the extension step is realigned to the regions around the seed, and the seed sequences in the alignment regions are removed from the seed table (Supplementary Figure S1).

RepeatScout performs the realignment for all seed sequences within the consensus sequence. This approach effectively reduces redundancy of repeat detection. However, because the realignment is conducted even for seeds not used for repeat detection, the sensitivity of detecting different repeat families that share the same seed may be reduced. Therefore, REPrise adopts a loose masking approach, which targets only the seeds used for repeat detection. Unlike masking in RepeatScout, this masking approach prevents the loss of sensitivity from inappropriate masking but introduces high redundancy among the detected repeat families. To mitigate this redundancy, REPrise performs CD-HIT on the identified repeat families after detecting all repeat families. According to the Wicker's 80/80/80 rule for repeat detection [32], we set the similarity threshold for CD-HIT at 80%.

Datasets and evaluation measures

We assessed the performance of REPrise using three genome datasets with curated repeat annotation: the rice, the simulation, and the complete human genome datasets. The first rice genome dataset [33] contains a manually curated repeat annotation generated using software tools such as RECON [15] and serves as a standard benchmark for performance validation of repeat annotation software [23]. The second simulation genome dataset was created using an existing TE insertion simulator that randomly incorporates sequences from a TE library into a long random sequence multiple times [24]. Mutations were introduced at varying rates, ranging from 5 mutations per 100 bases (a 5% mutation rate) to 45 mutations per 100 bases (a 45% mutation rate), depending on the simulation parameters. We also adjusted the ratio of substitutions to indels for the mutation as follows: 100:0, 80:20 and 60:40. We used a default TE library consisting of 20 TE families and inserted the TE sequences into a random sequence of 0.6 million nucleotides. This resulted in a sequence of approximately 4.6 million nucleotides for each parameter setting. The third dataset is complete human genome dataset is the T2T-CHM13 v2.0 [14] genome with the repeat annotation [1]. We used the thickStart and thickEnd columns in the BED file for the repeat annotation. We also masked tandem repeats using TANTAN [34]. Note that we did not mask tandem repeats in the rice genome dataset as the previous benchmark study did not use the masking tools.

We applied RepeatScout and REPrise to these genome datasets and identified the repeat families. Subsequently, we annotated the locations of the repeats within the genome using RepeatMasker and the identified repeat families. We then evaluated the performance of each software by comparing the software outputs with the curated annotation (Supplementary Figure S2(A)). We evaluated the overlap between the annotation and the software output at the nucleotide level and categorized the genomic regions into four groups: true positive (TP), true negative (TN), false positive (FP), and false negative (FN) (Supplementary Figure S2(B)). From these categories, we

calculated four evaluation measures: sensitivity, specificity, precision, and F-score.

Due to fragmented detection or low similarity among TE sequences, de-novo generated TE libraries may differ from the reference ones. To evaluate this, we directly compared the consensus sequences between the de novo generated TE library and the reference TE library by the method described in RepeatModeler2 paper (https://github.com/jmf422/TE_annotation) [26]. In addition to the overall comparison, we performed comparisons for each TE class. Each consensus in the reference library was classified into one of four categories: Perfect, Good, Present, or Not found, and the proportions of each category were examined. "Perfect" indicates that one sequence in the de novo generated library matches 95% in similarity and coverage to a consensus sequence in the reference library. "Good" refers to cases where multiple overlapping sequences in the de novo generated library, each with 95% similarity and total coverage. "Present" indicates that one or more sequences in the de-novo-generated library, with 80% similarity and and coverage.

We also used the complete human genome dataset to discover novel interspersed repeat family candidates. We first mapped the detected repeat families by REPrise to the complete human genome using RepeatMasker. We then defined genome regions that met the following two criteria as novel repeat regions: (i) no overlap with tandem repeats, centromere satellites [35], or segmental duplications [36], and (ii) less than 20% overlap with genes [37] or known repeat regions. For each repeat family, if 40% or more of the mapped regions were classified as novel repeat regions, we labeled the repeat family as a novel repeat family. We further validated the novel repeat family regions using the UCSC Genome Browser. When necessary, we retrieved transposon protein sequences from repeatmasker.org[10] and aligned them to the novel repeat regions using LAST [38]. We also performed phylogenetic analysis by aligning repeat region sequences with MAFFT version 7 [39], trimming the alignments with TrimAl [40], and constructing phylogenetic trees with IQ-TREE2 [41] when required.

Results

Evaluation in the rice genome dataset

We first explored the influence of seed length k on the repeat detection performance using the rice genome dataset. We set the gap open score as $o = 5$, the bandwidth as $b = 5$, and the gap extension score as $e = 1$ in this analysis. Supplementary Figure S3 shows the dependence of the detection performance on k of RepeatScout and REPrise with varying numbers of allowed mismatches d. We found that the detection performance was highly dependent on k and the optimal k also depended on d. These findings underscore the importance of selecting appropriate k when using RepeatScout and REPrise. In this study, we selected the k with the highest sensitivity. Specifically, for the rice genome dataset analysis, we employed $k = 15$ for RepeatScout and REPrise with $d = 0$, $k = 21$ for REPrise with $d = 1$, and $k = 25$ for REPrise with $d = 2$. Furthermore, we assessed the effect of b and e on the detection performance (Supplementary Figure S4). We fixed $o = 5$ in this analysis. We found that b does not substantially affect the performance, and thus we set $b = 5$ because smaller b should speed up the computations. Moreover, we set $e = 1$ because this value was the best in all e when $b = 5$.

Additionally, we investigated the dependence on the clustering programs using the rice genome dataset. We compared three clustering programs: CD-HIT, meshclust [42], and uclust [43]. The results showed that the number of clusters of CD-HIT was closest to 2431, the annotated number of families by Ou *et al.* (version 6.9.5.), and CD-HIT achieved the highest F1-score (Supplementary Table S1). Therefore, we used CD-HIT for clustering repeats.

We next compared the repeat detection performance of RepeatScout and REPrise (Table 1). Our results showed that REPrise demonstrated significantly improved sensitivity compared to RepeatScout, particularly with $d = 2$, where the sensitivity increased from 89.62% (RepeatScout) to 94.08% (REPrise). Furthermore, REPrise outperformed RepeatScout even when employing the exact seed ($d = 0$). Since the improvement from the affine gap scoring was minor in this analysis(Supplementary Figure S4), the loose masking likely played a significant role in enhancing the performance. In addition, when evaluating the impact of d on the performance, the sensitivity increased with a rise in d, indicating that a larger d is advantageous for detecting novel repeats.

These results suggest that the loose masking approach of REPrise likely contributed significantly to the substantial

Table 1 Software performance for the rice genome dataset

	Time[h:m:s]	Mem[GB]	Sensitivity	Specifiity	F-score
REPrise ($d = 0$)	0:49:12	8.67	93.40	93.72	**93.18**
REPrise ($d = 1$)	1:42:55	14.49	93.51	93.46	93.10
REPrise ($d = 2$)	35:17:53	14.20	**94.08**	92.66	92.98
RepeatScout	0:52:17	5.49	89.62	**95.82**	92.23

The bold values are the highest scores among the software. "Mem" represents the amount of memory used. Note that RepeatScout is a single-threaded computation, whereas REPrise is a 16 multi-threaded computation

enhancement in sensitivity. The comparison of REPrise with different d values suggests that inexact seeding also contributes to some degree of sensitivity improvement. The impact of affine gap scoring, though modest, still contributes positively to the overall sensitivity

Note that the manually curated annotation of the rice genome has limitations, and that the repeat regions labeled as the False Positive by REPrise are potentially unknown interspersed repeat regions. Therefore, even if the specificity is low, REPrise's ability to recognize non-repeat sequences should not be underestimated.

We also assessed the detection sensitivity across different repeat classes (Supplementary Table S2). Our results showed that REPrise consistently outperformed RepeatScout for all repeat classes. In addition, the detection sensitivity of REPrise improved for all repeat classes as d increased. In particular, the performance largely improved for non-LTR elements, which were difficult to detect sensitively due to their high variability [23].

The results of the direct comparison of the consensus sequences between the de novo generated TE library and the reference TE library are shown in Supplementary Figure S6. REPrise outperformed RepeatScout in detecting sequences present in the reference library. When analyzed by class, REPrise demonstrated superior performance, particularly for non-LTR repeats, as well as DNA transposons such as TIRs (Terminal Inverted Repeats) and Helitrons. However, REPrise with $d = 2$ failed to detect some repeats as "Perfect", suggesting that repeat sequences were identified in fragmented forms.

We finally evaluated the program runtimes (Table 1). The computations were performed on an Intel Xeon Gold 6130 (16 cores) 2.1GHz CPU with 192 GB of memory. Note that REPrise was multi-threaded during both the seed table construction and extension steps, utilizing 16 and two threads, respectively. Even when using the same exact seeding method, REPrise with $d = 0$ is expected to take longer than RepeatScout because REPrise used the suffix array instead of the hash table to construct the seed table. However, their computational speeds were comparable, indicating that the parallelization by multi-threading in REPrise efficiently contributes to the speed-up. In addition, the runtimes increased as d increased, with $d = 2$ taking roughly 42 times longer than $d = 0$. We also analyzed how varying the number of threads influenced the runtime in REPrise (Supplementary Figure S5). We verified that increasing the numer of threads to 16 reduced the runtime, but there was no significant change in the runtime beyond 16 threads.

Evaluation in the simulation genome dataset

We next investigated the effects of the different levels of similarity among repeat sequences on the detection performance using the simulation datasets. We first compared the detection performance for various k values. Then, we selected k values as 13 for RepeatScout, and 13, 15, and 18 for REPrise with d values of 0, 1, and 2, respectively (Supplementary Figure S7).

Figure 2 and Supplementary Figure S8 showed evaluation results for the simulation genome dataset. We verified that RepeatScout and REPrise had similar performance when the mutation rate was low, i.e. when the sequence similarity was high. Conversely, when the mutation rate increased to 30%, REPrise demonstrated enhanced performance with a larger d. This result suggests that the inexact seeding was effective in detecting repeats that contain many mutations. Furthermore, when mutations were predominantly substitutions, there was no performance difference between REPrise ($d = 0$) and RepeatScout. However, when the majority of the mutations were indels, REPrise ($d = 0$) showed better performance than RepeatScout. This result indicates that the affine gap scoring and the loose masking improved the detection of repeats abundant in indels.

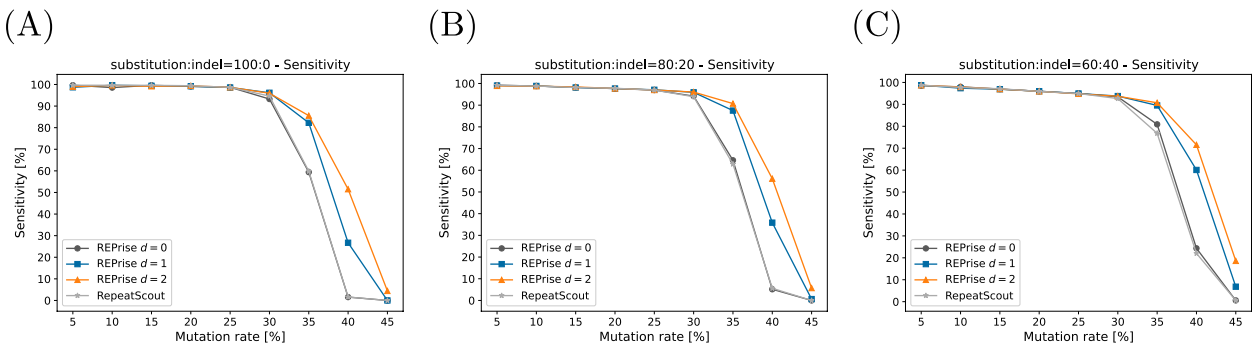

Fig. 2 Dependence of the detection sensitivity of RepeatScout and REPrise on the Mutation rate in the simulation genome dataset. The ratios of substitutions to indels were **A** 100:0, **B** 80:20, and **C** 60:40. The gray, black, blue, and orange lines represent RepeatScout, REPrise ($d = 0, 1, 2$), respectively. The x- and y-axes represent the mutation rate and the sensitivity, respectively

We also found that the detection performance of these programs decreased with increasing substitution rates to indels when the mutation rate was high. This may be because the high substitution rate to indels prevented sharing of seed sequences among repeats in the same repeat family. Conversely, when the substitution rate to indels was lower, shared seed sequences among repeats in the same repeat family are likely to be retained. However, the low substitution rates (high indel rates) had a drawback that the number of repeat families was overestimated, because the indels fragment a single repeat family into multiple distinct repeat families (Supplementary Figure S8).

In all simulation datasets, the number of identified families increased as the mutation rate increased, likely resulting from the fragmentation of detected repeat sequences. However, when the mutation rate was set extremely high, there was a noticeable decrease in the number of identified families, likely due to the reduced detection accuracy.

Analysis of the complete human genome dataset

We then applied REPrise to the complete human genome dataset. We selected 17, 21, and 23 as k for REPrise with $d = 0$, 1, and 2, respectively, after evaluating the detection performance among different k values (Supplementary Figure S9). Note that we could not employ RepeatScout for the human genome analysis because RepeatScout does not support application to long genomes. We validated that REPrise with $d = 2$ surpassed REPrise with $d = 0$ and $d = 1$ for all evaluation measures (Table 2).

We also explored the capability to identify novel repeat families for REPrise with $d = 2$. Consequently, we identified 17 novel repeat families (Supplementary Table S3). The novel repeat family with the highest proportion of novel repeat regions was novel_repeat_family-1 (NRF-1), where 100% of the repeat regions were novel. Upon investigation of these repeat regions, we found that they were located immediately upstream of each gene of the KRTAP9 gene family on chromosome 17 (Supplementary Figure S10). This result suggests that these repeat regions may be promoter regions that were duplicated during gene duplication in the KRTAP9 gene family. The novel repeat family with the second-highest proportion of novel repeat regions was NRF-2, with 71% of the 35 regions being novel. The majority of these regions were located on the Y chromosome, and parts of them were present in the long repetitive genomic regions newly identified by T2T-CHM13 (Supplementary Figure S11(A)). Given the abundance of tandem repeats on the Y chromosome, these regions may be part of tandem repeats, but they were not detected by existing tandem repeat detection tools, TRF [44] and WindowMasker [45] (Supplementary Figure S11(B)).

Table 2 Software performance for the complete human genome dataset

	Time[h:m:s]	Mem[GB]	Sensitivity	Specifiity	F-score
REPrise ($d = 0$)	5:57:46	36.68	76.96	91.85	84.09
REPrise ($d = 1$)	37:40:56	67.64	77.68	92.11	84.63
REPrise ($d = 2$)	693:34:27	151.67	**78.14**	**92.46**	**85.04**

The bold values are the highest scores among *d*. "Mem" represents the amount of memory used. The computations were performed on an Intel Xeon Gold 6148 Base 2.4GHz CPU with 32 parallel cores and 12GB memory per core

In the identified novel repeat families, two families may be unidentified subfamilies of existing LTR repeat families. The LTR regions are repeat regions that originally existed in pairs flanking the retroviral genome in the same orientation [46]. Although some LTR region are now isolated by the recombination, this characteristic LTR structure is retained in many genome regions. he first LTR subfamily candidate was NRF-10, which represents the LTR portion of an LTR retrotransposon and is present as a solo LTR or part of an intact LTR element. NRF-10 seemed to be a subfamily of the LTR19 family, because NRF-10 frequently appeared adjacent to fragments of the LTR19 family members (Fig. 3A, Supplementary Figure S12). In one case, NRF-10 was adjacent to an intact LTR-19 family member with two LTR regions flanking the internal sequence, suggesting that NRF-10 derived from an LTR element (Supplementary Figure S12).

To test this hypothesis, we conducted two additional analyses. First, we retrieved the protein sequences of HERVFH19, an endogenous retrovirus related to the LTR19 family [49], from repeatmasker.org [10] and aligned the ±10kb regions of NRF-10 using LAST [38]. As a result, HERVFH19 protein sequences were detected in 12 out of 31 NRF-10 regions. Notably, these elements were found only on one side of the NRF-10 regions (Supplementary Figure S13). This observation suggests that NRF-10 elements occur at the termini of HERVFH19, consistent with the characteristics of LTR sequences. Second, we conducted a phylogenetic tree analysis of the LTR19 family and NRF-10. Specifically, we first extracted all annotated sequences of the LTR19 family and NRF-10 using SeqKit [50]. When NRF-10 was found adjacent to the LTR19 family within a 500-base span, these sequences were concatenated into a single sequence using bedtools [51]. We used MER50C as the outgroup because MER50C diverged earlier than LTR19A, B, and C [52]. As a result, sequences

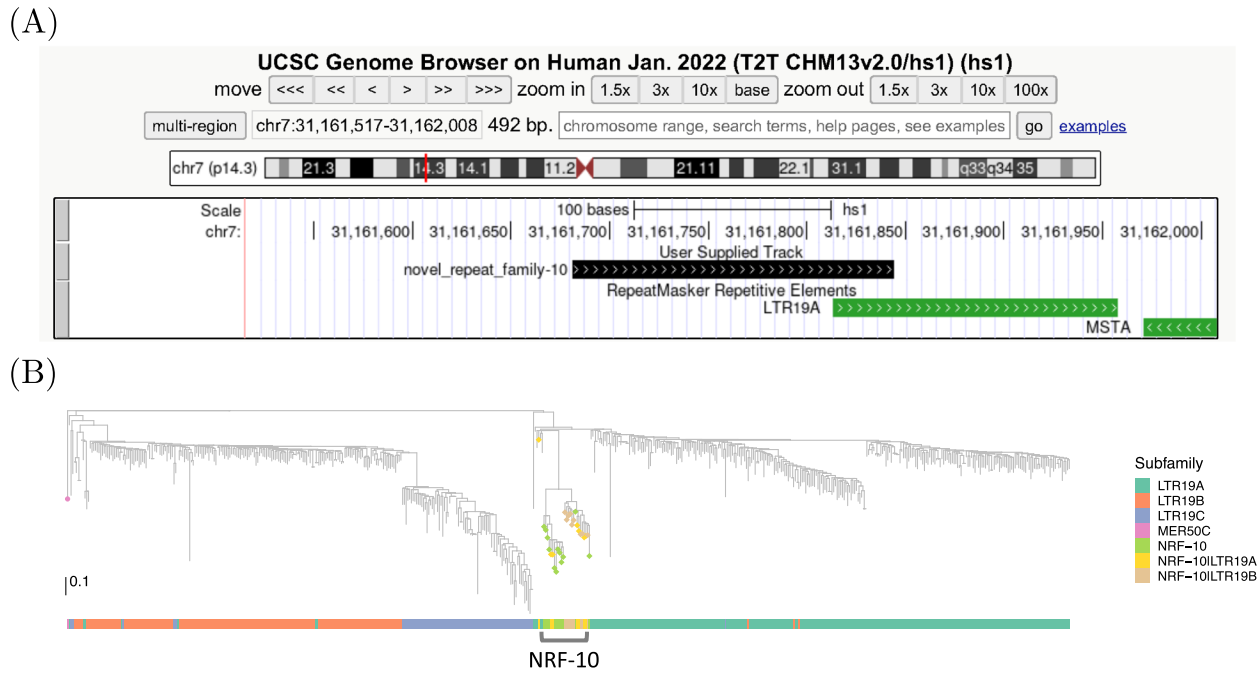

Fig. 3 **A** An example of NRF-10 ('novel_repeat_family-10' in the figure) region with RepeatMasker Repetitive Elements displayed on the UCSC genome browser. **B** A phylogenetic tree of annotated regions of NRF-10 and LTR19 subfamilies. NRF-10|LTR19-A and NRF-10|LTR19-B are concatenated sequences of NRF-10 and LTR-19. The phylogenetic tree was visualized using ggtree [47] and ggtreeExtra [48]

containing NRF-10 form a subtree within the LTR19 family. These two results suggest that NRF-10 represents a novel subfamily of the LTR19 family that is associated with HERVFH19.

The second LTR subfamily candidate was NRF-13, as many sequences from this repeat family overlapped with the LTR45 family (Supplementary Figure S14(A,B)). In addition, like NRF-10, there were also an example of LTR-specific structures forming in NRF-13 as well as NRF-10 (Supplementary Figure S14(C)). Therefore, we conducted a phylogenetic tree analysis for NRF-13. However, given the absence of a suitable outgroup for the LTR45 family, we established an unrooted phylogenetic tree (Supplementary Figure S15). We found that sequences encompassing NRF-13 also clustered into a distinct subtree, potentially representing another subfamily within the LTR45 family.

In this study, to investigate the usefulness of REPrise in identifying highly novel repeat families, we adopted strict criteria for their identification. Therefore, although we discovered some novel repeat families, we may have missed unidentified repeat subfamilies that are largely consistent with known repeat families but possess distinct regions. A more detailed analysis of individual repeat families detected by REPrise should reveal further novel repeat subfamilies.

Discussions

In this study, we developed REPrise, a highly sensitive tool for de novo detection of interspersed repeats. The significant advantage of REPrise is that it can be applied to the entire large genome sequences, such as the human genome. Currently, the widely used RepeatModeler2 pipeline for de novo repeat library construction relies on the internal execution of RepeatScout. RepeatModeler2 samples approximately 400MB of sequences during de novo repeat library creation, constituting only about 13% of the human genome. Due to the non-uniform distribution of interspersed repeats, there is a potential risk of overlooking certain repeat sequences, particularly in species with large genomes. In contrast, REPrise is inherently designed to handle large sequences and exhibits linear time complexity in relation to sequence length, addressing these limitations in current methodologies.

One of the important issues with REPrise is its computational cost. For example, running the human genome with $d = 2$ requires approximately 700 hours (Table 2). Therefore, future development to make REPrise faster is an essential research issue. As the bottleneck in REPrise's runtime lies in the repeated search for inexact seed, potential improvements include optimizing seed search algorithms and storing intermediate results. On the other hand, since repeat annotation is typically performed only

once after genome assembly, the long execution time may not be a critical limitation for most users.

The other issue with REPrise, similar to other de novo repeat detection software [24], is that it cannot capture the detailed structure of repeats due to challenges such as fragmentation (Supplementary Figure S6). Addressing this issue requires the use of TE curation pipelines [26], or performing manual curation [53]. REPrise can be used as part of such pipelines or manual curation, and its high sensitivity may help enhance their quality.

REPrise has multiple parameters, and the proper setting is crucial for achieving high detection performance. In particular, the seed length k has a significant impact on the detection performance. In this study, we applied REPrise with various k values to genome data and selected the k value that achieved the highest performance. However, in practical setting, k must be automatically determined. In RepeatScout, the default k was set using the formula: $k = \log_4 |T| + 1$. This length was chosen such that the expected occurrence of a seed sequence within a random genome sequence is less than one. Based on this concept, we set the default k in REPrise as follows. Let l be the smallest natural number for which $1 > \frac{|T| \cdot \Sigma_{d'=0}^{d} \binom{l}{d'} 3^{d'}}{4^l}$, and then set $k = l + 1$. When comparing the automatically determined k value with the empirically determined k value, the two values were similar (Supplementary Table S4). This result suggests that our method of automatically determining k holds some level of validity. Exploring a more appropriate automatic method for determining k remains an essential topic for future research.

We envisioned two strategies for further enhancing the performance of REPrise. The first focuses on refining the seed design. Although the inexact seeding has improved detection sensitivity, the computational runtimes have also significantly increased. Therefore, improving computational speed is a critical issue. One possible solution is to adopt sparse seeding, a method commonly used in the sequence alignment, particularly for mapping long reads [54]. In this design, only a subset of subsequences in the genome serve as candidate seeds, rather than utilizing all subsequences. While this approach could drastically reduce the computation time, it may also diminish the detection sensitivity. Another direction for refining seed design is accounting for indels to further improve sensitivity. Recently, the 'strobemer' has been proposed as a seed design for efficient handling of indels in sequence alignment [55]. The strobemer is a combination of multiple short k-mers and can account for potential insertions between two k-mers. The application of the strobemer to interspersed repeat detection may lead to the discovery of further novel repeat family candidates.

The second is enhancing the estimation accuracy of the number of repeat families. Our simulation experiments revealed that both RepeatScout and REPrise overestimated the number of families when the number of indels was high. Because many interspersed repeats in empirical genomes have nested structures or large indels [1], the current estimates for the number of repeat families may be inflated. One feasible solution is to merge repeat families based on sequences surrounding the detected repeats. For instance, Red [18] and RepLoc [20] determine repeat positions in the genome using k-mers without identifying the repeat families, and integrating these tools into REPrise could improve the accuracy of family number estimates.

Conclusion

REPrise is a de novo repeat detection software based on the seed-and-extend approach, with enhancements such as inexact seeding, affine gap scoring, and loose masking to improve sensitivity. REPrise showed better repeat detection performance than RepeatScout on both empirical and simulation genome datasets, especially for repeats with many sequence mutations. Additionally, as a practical application of REPrise, we identified several novel repeat families in the complete human genome T2T-CHM13 v2.0. REPrise should be a valuable tool for comprehensive repeat annotation, especially in genomes lacking well-curated repeat databases.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s13100-025-00353-0.

Supplementary Material 1.

Acknowledgements

We thanks to members of Hamada Laboratory, especially Kento Kubo and Masato Kosuge for their valuable comments. A.T. is grateful to Sigehiro Kuraku (National Institute of Genetics) for insightful discussions. A.T. thanks to Naoki Konno (Department of Biological Science, Graduate School of Science, The University Tokyo) for technical supports in phylogenetic tree creation. Computations were partially performed on the NIG supercomputer at ROIS National Institute of Genetics.

Authors' contributions

T.F. and H.M. designed and conceived this study. D.N. and A.T. implemented the software. Y.I. and A.T. developed the evaluation pipeline of the software. A.T. investigated the software performance. T.F. and H.M. supervised the study. A.T. and T.F. prepeared the draft manuscript. H.M. reviewed and edited the manuscript.

Funding

This work was supported by JSPS KAKENHI (Grant Numbers: JP23KJ2044 to A.T.; JP23K16997 to T.F.; 22H04925, 20H00624 and 23H00509 to M.H.) and AMED (Grant Numbers: JP22ama121055, JP21ae0121049 and JP21gm0010008 to M.H.).

Data availability

Rice genome and its annotations: https://github.com/oushujun/EDTA. Simulation sequence generation: https://github.com/IOB-Muenster/denovoTE-eval. Complete human genome and its annotations: https://sites.google.com/ucsc.edu/t2tworkinggroup.

Code availability
The source code of REPrise and scripts for evaluation are freely available at https://github.com/hmdlab/REPrise.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
All authors consent to the publication of this manuscript.

Competing interests
The authors declare no competing interests.

Received: 6 June 2024 Accepted: 17 March 2025
Published online: 03 April 2025

References

1. Hoyt SJ, Storer JM, Hartley GA, Grady PG, Gershman A, et al. From telomere to telomere: The transcriptional and epigenetic state of human repeat elements. Science. 2022;376(6588):eabk3112.
2. The International Wheat Genome Sequencing Consortium (IWGSC), Appels R, Eversole K, Stein N, Feuillet C, Keller B, et al. Shifting the limits in wheat research and breeding using a fully annotated reference genome. Science. 2018;361(6403):eaar7191. https://doi.org/10.1126/science.aar7191.
3. Attig J, Agostini F, Gooding C, Chakrabarti AM, Singh A, Haberman N, et al. Heteromeric RNP assembly at LINEs controls lineage-specific RNA processing. Cell. 2018;174(5):1067–81.
4. Chishima T, Iwakiri J, Hamada M. Identification of transposable elements contributing to tissue-specific expression of long non-coding RNAs. Genes. 2018;9(1):23.
5. Zeng C, Onoguchi M, Hamada M. Association analysis of repetitive elements and R-loop formation across species. Mob DNA. 2021;12(1):1–11.
6. Dodsworth S, Chase MW, Kelly LJ, Leitch IJ, Macas J, Novak P, et al. Genomic repeat abundances contain phylogenetic signal. Syst Biol. 2015;64(1):112–26.
7. Lerat E. Identifying repeats and transposable elements in sequenced genomes: how to find your way through the dense forest of programs. Heredity. 2010;104(6):520–33.
8. Bao W, Kojima KK, Kohany O. Repbase Update, a database of repetitive elements in eukaryotic genomes. Mob DNA. 2015;6(1):11. https://doi.org/10.1186/s13100-015-0041-9.
9. Storer J, Hubley R, Rosen J, Wheeler TJ, Smit AF. The Dfam community resource of transposable element families, sequence models, and genome annotations. Mob DNA. 2021;12(1):1–14.
10. Smit AFA, Green P. RepeatMasker Open-4.0. 2013-2015.
11. Xu Z, Wang H. LTR_FINDER: an efficient tool for the prediction of full-length LTR retrotransposons. Nucleic Acids Res. 2007;35(suppl_2):W265–8.
12. Szak ST, Pickeral OK, Makalowski W, Boguski MS, Landsman D, Boeke JD. Molecular archeology of L1 insertions in the human genome. Genome Biol. 2002;3:1–18.
13. Lewin HA, Richards S, Aiden EL, Allende ML, Archibald JM, Bálint M, et al. The Earth BioGenome Project 2020: Starting the clock. Proc Natl Acad Sci. 2022;119(4):e2115635118. https://doi.org/10.1073/pnas.2115635118.
14. Nurk S, Koren S, Rhie A, Rautiainen M, Bzikadze AV, et al. The complete sequence of a human genome. Science. 2022;376(6588):44–53.
15. Bao Z, Eddy SR. Automated de novo identification of repeat sequence families in sequenced genomes. Genome Res. 2002;12(8):1269–76.
16. Gu W, Castoe TA, Hedges DJ, Batzer MA, Pollock DD. Identification of repeat structure in large genomes using repeat probability clouds. Anal Biochem. 2008;380(1):77–83. https://doi.org/10.1016/j.ab.2008.05.015.
17. Flutre T, Duprat E, Feuillet C, Quesneville H. Considering transposable element diversification in de novo annotation approaches. PLoS ONE. 2011;6(1):e16526.
18. Girgis HZ. Red: an intelligent, rapid, accurate tool for detecting repeats de-novo on the genomic scale. BMC Bioinformatics. 2015;16(1):227. https://doi.org/10.1186/s12859-015-0654-5.
19. Schaeffer CE, Figueroa ND, Liu X, Karro JE. phRAIDER: Pattern-Hunter based Rapid Ab Initio Detection of Elementary Repeats. Bioinformatics. 2016;32(12):i209–15. https://doi.org/10.1093/bioinformatics/btw258.
20. Feng C, Dai M, Liu Y, Chen M. Sequence repetitiveness quantification and de novo repeat detection by weighted k-mer coverage. Brief Bioinforma. 2020;22(3):bbaa086. https://doi.org/10.1093/bib/bbaa086.
21. Price AL, Jones NC, Pevzner PA. De novo identification of repeat families in large genomes. Bioinformatics. 2005;21(suppl_1):i351–i358. https://doi.org/10.1093/bioinformatics/bti1018.
22. Saha S, Bridges S, Magbanua ZV, Peterson DG. Empirical comparison of ab initio repeat finding programs. Nucleic Acids Res. 2008;36(7):2284–94. https://doi.org/10.1093/nar/gkn064.
23. Ou S, Su W, Liao Y, Chougule K, Agda JRA, Hellinga AJ, et al. Benchmarking transposable element annotation methods for creation of a streamlined, comprehensive pipeline. Genome Biol. 2019;20(1):275. https://doi.org/10.1186/s13059-019-1905-y.
24. Rodriguez M, Makałowski W. Software evaluation for de novo detection of transposons. Mob DNA. 2022;13(1):1–14.
25. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. J Mol Biol. 1990;215(3):403–10. https://doi.org/10.1016/S0022-2836(05)80360-2.
26. Flynn JM, Hubley R, Goubert C, Rosen J, Clark AG, Feschotte C, et al. RepeatModeler2 for automated genomic discovery of transposable element families. Proc Natl Acad Sci. 2020;117(17):9451–7. https://doi.org/10.1073/pnas.1921046117.
27. Ma B, Tromp J, Li M. PatternHunter: faster and more sensitive homology search. Bioinformatics. 2002;18(3):440–5.
28. Gotoh O. An improved algorithm for matching biological sequences. J Mol Biol. 1982;162(3):705–8.
29. Fu L, Niu B, Zhu Z, Wu S, Li W. CD-HIT: accelerated for clustering the next-generation sequencing data. Bioinformatics. 2012;28(23):3150–2.
30. Nong G, Zhang S, Chan WH. Linear Suffix Array Construction by Almost Pure Induced-Sorting. In: 2009 Data Compression Conference. 2009. pp. 193–202. https://doi.org/10.1109/DCC.2009.42.
31. Shrestha AMS, Frith MC, Horton P. A bioinformatician's guide to the forefront of suffix array construction algorithms. Brief Bioinforma. 2014;15(2):138–54.
32. Wicker T, Sabot F, Hua-Van A, Bennetzen JL, Capy P, Chalhoub B, et al. A unified classification system for eukaryotic transposable elements. Nat Rev Genet. 2007;8(12):973–82.
33. International Rice Genome Sequencing Project, Sasaki T. The map-based sequence of the rice genome. Nature. 2005;436(7052):793–800.
34. Frith MC. A new repeat-masking method enables specific detection of homologous sequences. Nucleic Acids Res. 2011;39(4):e23.
35. Altemose N, Logsdon GA, Bzikadze AV, Sidhwani P, Langley SA, Caldas GV, et al. Complete genomic and epigenetic maps of human centromeres. Science. 2022;376(6588):eabl4178.
36. Vollger MR, Guitart X, Dishuck PC, Mercuri L, Harvey WT, Gershman A, et al. Segmental duplications and their variation in a complete human genome. Science. 2022;376(6588):eabj6965.
37. Pruitt KD, Brown GR, Hiatt SM, Thibaud-Nissen F, Astashyn A, Ermolaeva O, et al. RefSeq: an update on mammalian reference sequences. Nucleic Acids Res. 2014;42(D1):D756–63.
38. Kiełbasa SM, Wan R, Sato K, Horton P, Frith MC. Adaptive seeds tame genomic sequence comparison. Genome Res. 2011;21(3):487–93.
39. Katoh K, Standley DM. MAFFT multiple sequence alignment software version 7: improvements in performance and usability. Mol Biol Evol. 2013;30(4):772–80.
40. Capella-Gutiérrez S, Silla-Martínez JM, Gabaldón T. trimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses. Bioinformatics. 2009;25(15):1972–3.
41. Minh BQ, Schmidt HA, Chernomor O, Schrempf D, Woodhams MD, Von Haeseler A, et al. IQ-TREE 2: new models and efficient methods for phylogenetic inference in the genomic era. Mol Biol Evol. 2020;37(5):1530–4.
42. Girgis HZ. MeShClust v3. 0: high-quality clustering of DNA sequences using the mean shift algorithm and alignment-free identity scores. BMC Genomics. 2022;23(1):423.

43. Edgar RC. Search and clustering orders of magnitude faster than BLAST. Bioinformatics. 2010;26(19):2460–1.
44. Benson G. Tandem repeats finder: a program to analyze DNA sequences. Nucleic Acids Res. 1999;27(2):573–80. https://doi.org/10.1093/nar/27.2.573.
45. Morgulis A, Gertz EM, Schäffer AA, Agarwala R. WindowMasker: window-based masker for sequenced genomes. Bioinformatics. 2005;22(2):134–41. https://doi.org/10.1093/bioinformatics/bti774.
46. Thomas J, Perron H, Feschotte C. Variation in proviral content among human genomes mediated by LTR recombination. Mob DNA. 2018;9:1–15.
47. Xu S, Li L, Luo X, Chen M, Tang W, Zhan L, et al. Ggtree: a serialized data object for visualization of a phylogenetic tree and annotation data. IMeta. 2022;1(4):e56.
48. Xu S, Dai Z, Guo P, Fu X, Liu S, Zhou L, et al. ggtreeExtra: compact visualization of richly annotated phylogenetic data. Mol Biol Evol. 2021;38(9):4039–42.
49. Kojima KK. Human transposable elements in Repbase: genomic footprints from fish to humans. Mob DNA. 2018;9(1):2.
50. Shen W, Le S, Li Y, Hu F. SeqKit: a cross-platform and ultrafast toolkit for FASTA/Q file manipulation. PLoS ONE. 2016;11(10):e0163962.
51. Quinlan AR, Hall IM. BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics. 2010;26(6):841–2.
52. Benachenhou F, Blikstad V, Blomberg J. The phylogeny of orthoretroviral long terminal repeats (LTRs). Gene. 2009;448(2):134–8.
53. Goubert C, Craig RJ, Bilat AF, Peona V, Vogan AA, Protasio AV. A beginner's guide to manual curation of transposable elements. Mob DNA. 2022;13(1):7.
54. Li H. Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics. 2018;34(18):3094–100.
55. Sahlin K. Effective sequence similarity detection with strobemers. Genome Res. 2021;31(11):2080–94.

Publisher's Note