Interdisciplinary Sciences: Computational Life Sciences (2022) 14:141–150
https://doi.org/10.1007/s12539-021-00476-x

ORIGINAL RESEARCH ARTICLE

A Hierarchical Error Correction Strategy for Text DNA Storage

Xiangzhen Zan[1] · Xiangyu Yao[1] · Peng Xu[1] · Zhihua Chen[1] · Lian Xie[2] · Shudong Li[3] · Wenbin Liu[1]

Received: 10 June 2021 / Revised: 20 August 2021 / Accepted: 22 August 2021 / Published online: 31 August 2021

Abstract
DNA storage has been a thriving interdisciplinary research area because of its high density, low maintenance cost, and long durability for information storage. However, the complexity of errors in DNA sequences including substitutions, insertions and deletions hinders its application for massive data storage. Motivated by the divide-and-conquer algorithm, we propose a hierarchical error correction strategy for text DNA storage. The basic idea is to design robust codes for common characters which have one-base error correction ability including insertion and/or deletion. The errors are gradually corrected by the codes in DNA reads, multiple alignment of character lines, and finally word spelling. On one hand, the proposed encoding method provides a systematic way to design storage friendly codes, such as 50% GC content, no more than 2-base homopolymers, and robustness against secondary structures. On the other hand, the proposed error correction method not only corrects single insertion or deletion, but also deals with multiple insertions or deletions. Simulation results demonstrate that the proposed method can correct more than 98% errors when error rate is less than or equal to 0.05. Thus, it is more powerful and adaptable to the complicated DNA storage applications.

Keywords DNA storage · Insertion · Deletion · Substitution · Robust codes

1 Introduction

The coming of big data era presents an increasing challenge for traditional storage technologies, such as optical disks, magnetic tapes, flashes memories, and other storage devices [1–3]. It is estimated that the global data will reach 1.75×10^{14} GB by 2025 [4], and it seems even if magnetic tape might not be sufficient in the next few decades. However, it is theoretically possible to store all the information recorded throughout human history in a space roughly the size of a double garage. Since DNA molecules have the characters of high density, low maintenance cost, and long durability [5–7], it is necessary and urgent to develop practical DNA storage technology to store the increasing massive amount of data.

The processes of DNA storage include four steps: encoding information into to DNA sequences, synthesize corresponding DNA strands, amplifying by polymerase chain reaction (PCR), and reading by sequencing technologies. During the storage process, the errors in DNA synthesis, proliferation, and sequencing, hinders its application in reliable information storage. It is estimated that the error rate in the second-generation synthesis and sequencing technologies is about 1–2% [8] and the third-generation technologies may have high error rate (about 10–15%) [9]. Base substitutions, insertions and deletions (indels) in the process of DNA molecular reaction will lead to errors. Such composite errors are beyond the capacity of traditional error correction codes (ECCs), including Hamming code [10, 11], BCH code [12, 13], Reed-Solomon code [14–17] and LDPC code [18–20], as they are designed to deal with only substitution errors. The wrong insertion and deletion of base not only increase the complexity of the error, but also lead to length inconsistency of many DNA sequences. The common sequencing technologies, such as nanopore technology, [21] usually results in over 88% reads having incorrect lengths. To deal with this situation, some works applying RS code or fountain code generally discard the incorrect length sequences [14, 16, 17, 21–23]. In addition,

✉ Wenbin Liu
wbliu6910@gzhu.edu.cn

[1] Institution of Computational Science and Technology, Guangzhou University, Guangzhou 510006, China

[2] Institution of Huangpu Research, Guangzhou University, Guangzhou 510006, China

[3] Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China

the reading process by sequencing is essentially a sampling process which usually results in the uneven distribution of sequences. Discarding the sequences with incorrect length may further deteriorate the data recovering process from the under-sampled sequences. Therefore, it is necessary to develop a new method to reduce and correct the complex errors inherent in DNA storage processes.

To reduce the reaction errors in synthesis, PCR, and sequencing processes, some constraints such as GC content of 45–55%, homopolymer-run length up to 3 and no potential of secondary structures in DNA strands are introduced [22, 24]. Church et al. [25] devised an one bit per base (A or C for zero, G or T for one) encoding strategy to avoid extreme GC content, homopolymers, or secondary structure. Goldman et al. [26] developed a 3-base rotation encoding scheme to ensure no homopolymers. Erlich et al. [22] and Anavy et al. [23] directly rejected these DNA sequences violating the homopolymer or GC content constraints in the construction of random DNA droplets. Some works [15, 21] transformed the binary stream by an XOR operation with a pseudorandom binary sequence before encoding them into DNA sequences to comply with biological constraints.

The existing error correction methods essentially add some redundancy to the original information strands. Goldman et al. [26] adopted a fourfold redundancy which encoded one sub-sequence in four consecutive strands. Bornholt et al. [27] proposed to generate a third redundant sequence based on the XOR operation of any two original sequences. Erlich et al. [22] introduced the fountain code into DNA storage which stored the redundant binary strands generated by the bit-wise addition of a random subset of the original binary strands. Linda et al. [14] proposed a novel outer-inner coding strategy to recover the missing sequences by the outer code redundancy, and correct erasures and substitutions within a sequence by the inner code redundancy. Anavy et al. [23] proposed composite DNA letters to reduce the errors in data recovery. Wang et al. [28] described a repeat accumulate coding strategies to cope with the missing sequences by bit-wise modulo-2 sum of the parity packet and source packets. Zhang et al. [29] proposed a quaternary Reed-Solomon coding to ensure the reliability of channel transmission.

Recently, some works focus on dealing with the insertion and deletion errors in DNA storage. Xue et al. [30] proposed a systematic code based on the Levenshtein code which could correct a single substitution, deletion or insertion error by adding some parity bits in the original message sequence. William et al. [31] proposed HEDGES (Hash Encoded, Decoded by Greedy Exhaustive Search) error-correcting code to repair the three basic types of DNA errors within one sequence. Lifu Song et al. [32] developed an algorithm named DBG-GPS (de Brujin graph-based greedy path searching) which could correct indels and substitutions.

As a large part of the archival data, the storage of text information in DNA received less attention. Zhong et al. [33] stored "The Analects of Confucius" by the quaternary code of Chinese (0-A, 1-T, 2-C, 3-G). However, the average coverage of sequencing is about 2700X to recover the original information as they did not apply any error correction method. Lee et al. [34] tried to store the "Universal Declaration of Human Rights" in plasmid. The erroneous nucleotide was corrected using the overlap extension PCR which is low efficiency and only feasible to very low error situations. In fact, there is no essential difference between these works as other binary data storage as they did not take advantage of the characteristics of text data.

In this paper, a hierarchical error correction strategy is proposed to store English text by DNA. First, a robust code book for common symbols in English text is designed, which can detect and correct both insertion/deletion and substitution of the nucleotides within a DNA coding fragment. Then, some of the remaining errors can be corrected by multiple alignments of character sequences. Finally, word spelling check is applied to further recover the file content. Computer simulation results show that the proposed error correction strategy can determine the most probabilistic file contents with minimal errors, thus suitable in large-scale DNA data storage for English text.

The remainder of this paper is organized as follows: the DNA storage scheme for English text is described in detail in Sect. 2; our simulation results of DNA storage and optimization results are presented in Sect. 3; finally, the conclusion is given in Sect. 4.

2 Materials and Methods

2.1 Encoding Method

Concerning the storage of English text, we adopt the code book strategy to encode the common characters by 30 six-base strings, which is different from previous works to encode 01 binary strings directly with A, T, C and G. Table 1 shows the code book for 26 letters, the digits from '0' to '9', and other 25 punctuations. The four shadowed codes in the bottom of Table 1 are uniquely assigned to the frequently used space character, and three shift keys to upper-case letter, digit and punctuation, respectively. This code book can encode $26 \times 3 + 4 = 82$ characters at most, which corresponds to a code density 1.06 bit/base.

As the GC content of each code is 50% (except for "TGCATA" and "GTATGA"), it assures that the encoded sequences have a uniform GC content. In addition, the homopolymer-run within any code and their concatenation is at most 2. Therefore, the code strategy can prohibit the homopolymer subsequences larger than 2. Furthermore, the

Table 1 The code book for text characters

DNA code	Alphabet	Punctuation	Digit	DNA code	Alphabet	Punctuation	Digit
TAACCG	a	@	4	ACACAC	l		6
TAAGGC	p	¥		ACTCTG	i	"	5
ATCACG	e	$	2	TCAGAG	j	%	
ATGAGC	y	,	8	ACAGGT	x	}{	
ATGGAG	g	*		ACTGCA	f	~	9
TACCAC	k	/		AGACCT	s	+	0
ATCCGT	b	()		TCTCGT	h	-	
ATGCCA	v	:		TCGAAC	c	?	1
TAGCGA	r	'	3	ACGACT	o	!	
TAGGCT	t	[]		CATTCG	z	&	
ACATCG	w	;		CTACAG	q	=	
ACTTGC	n			CAACGT	d	__	
TCTACG	m	Enter		CAGACA	u	#	7
TGCATA	Upper-case shift			GTATGA	Punctuation shift		
CTTGTC	Digit shift			CGGTAT	Space character		

undesired secondary structure in DNA sequences is related to many factors, such as the codes used, the assignment of the codes to the characters, the word composition in English, and the length of the DNA sequences. We apply a random search process to obtain the suboptimal assignment shown in Table 1. The detailed processes are discussed in the Sect. 3.

The Hamming distance between two sequences with the same length is the number of positions where the characters are not the same. The minimal Hamming distance between the 30 codes in Table 1 is 3. If the Hamming distance between any six-base sequence and a code c in Table 1 is one, then the Hamming distance between it and other codes will be at least 2. This means the observed six-base sequence may be resulted from one substitution error from code c, and it should be corrected to c. In information theory, it is a general principle that a Hamming distance $d = 2t + 1$ has the ability to correct t errors. Therefore, the code book in Table 1 has the ability to detect errors and correct 1-base errors ($d = 3, t = 1$). In addition, the minimal Hamming distance between the four codes in lower part of Table 1 is 5 and the minimal Hamming distance between them and the other 26 codes is 4. This provides them a more powerful error correction ability than the other 26 codes.

2.2 Decoding Method

The base insertion and deletion make the errors in DNA storage more complicated than that in traditional communication processes. Based on the capability of error corrections for codes in Table 1, our idea is to gradually correct the errors in three steps. First, correct a large part of errors in the sequenced reads based on the codes in Table 1. Second, translate reads into character lines and cluster them to form line clusters, then further correct some of the uncorrected errors based on multiple sequence alignment. Finally, use the current available spelling check software to perform word correction as used in Microsoft Word application. In DNA storage, the final consensus sequence is generally determined from multiple reads which may come from a unique encoding sequence. Therefore, the three-level strategy not only provides the process of error detection and correction, but obtains more reliable reads for the subsequent reads clustering as the first level corrects the errors in the index to improve its reliability. Figure 1B shows the workflow of the proposed three-level error correction.

Since the accurate detection of errors is the key to correct them, we first study the influence of one-base error in a sequence. Given a sequence with errors, we first cut it into 6-base slices (see Fig. 2A) and then calculate the minimal Hamming distance of these slices with the legal codes in Table 1. Figure 2A shows the minimal Hamming distance list of the original sequence and that with one-base error in the second code position, respectively. It is obvious that substitution has no effect on the following Hamming list, while either insertion or deletion can result in a sudden jump of the minimal Hamming distance in the following list. That is, insertion and/or deletion can propagate their effect while substitution does not affect the following list. This phenomenon is easy to understand as the legal codes in Table 1 are less than 1% of the 6-base sequences (total number is 4096). It is highly impossible that the base shift caused by insertion or deletion may meet a legal code or that is very similar to it. Therefore, a nonzero minimal Hamming distance is a strong signal to detect the occurrence of errors. Furthermore, if the type of errors, such as one or two base insertions/deletions, can be correctly determined, then we can remove its effect on the subsequent neighboring slices. That is, the value of

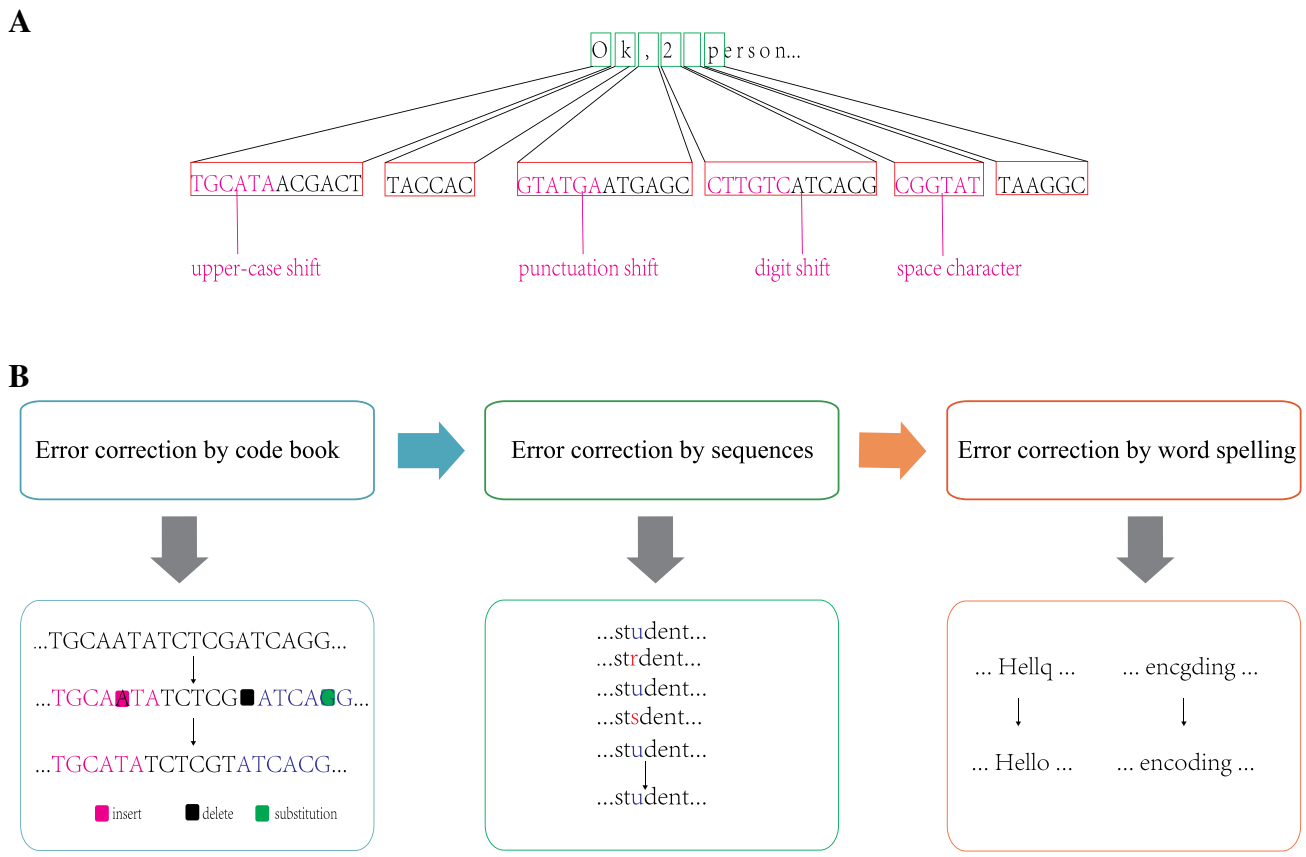

Fig. 1 The schematic diagrams for encoding and three-level error corrections. **A** is for encoding. **B** is the work flow for three-level error corrections

minimal Hamming distance in the subsequent neighboring slices will drop significantly when they have little other errors (see the values on the six lines in Fig. 2B). Therefore, we define an index δ to measure the difference of the minimal Hamming distance list before and after a possible error correction. It is defined as

$$\delta = h^0 - h' \tag{1}$$

where h^0 denotes the sum of the minimal Hamming distance of the k subsequent slices and h' is the one given a specific error type. Obviously, δ is zero for any substitution errors.

For simplicity, we only consider one or two base substitutions, insertions and deletions. The composite error of one insertion and one deletion is treated as two substitutions, one substitution and one insertion as one insertion, one substitution and one deletion as one deletion. Given a 6-base slice s where an error is detected, we first calculate the δ of each error type and select the type with the minimal δ for further consideration. If there are multiple options, the order of choice is substitution, insertion and deletion. Second, enumerate all possible corrections and select the one with minimal edit distance to correct s.

Based on the assumption that error rate is relatively low, the above process can correct most of the one-base substitution, insertion or deletion on slices. Although other composite errors may not be corrected properly, the determination of error type may help to eliminate the base shift effects caused by them, and thus alleviate the complexity of the following correction processes.

In sum, the first-level error correction process can be described as followed:

Step 1: cut a read into 6-base slices and calculate their minimal Hamming distances.

Step 2: detect the first error slice and determine the error type by the minimal δ.

Step 3: correct slice s by the minimal edit distance.

Step 4: repeat until to the end.

The second level refers to correct some of unrecovered or wrongly corrected codes in the first level by multiple alignment of reads coming from the same sequences. To improve the alignment effect, a weight is assigned for each read, which indicates the percentage of the correct codes. The correct codes include the uncontaminated codes and the corrected ones by one edit distance. First, the reads are translated into character lines. Second, group the lines by

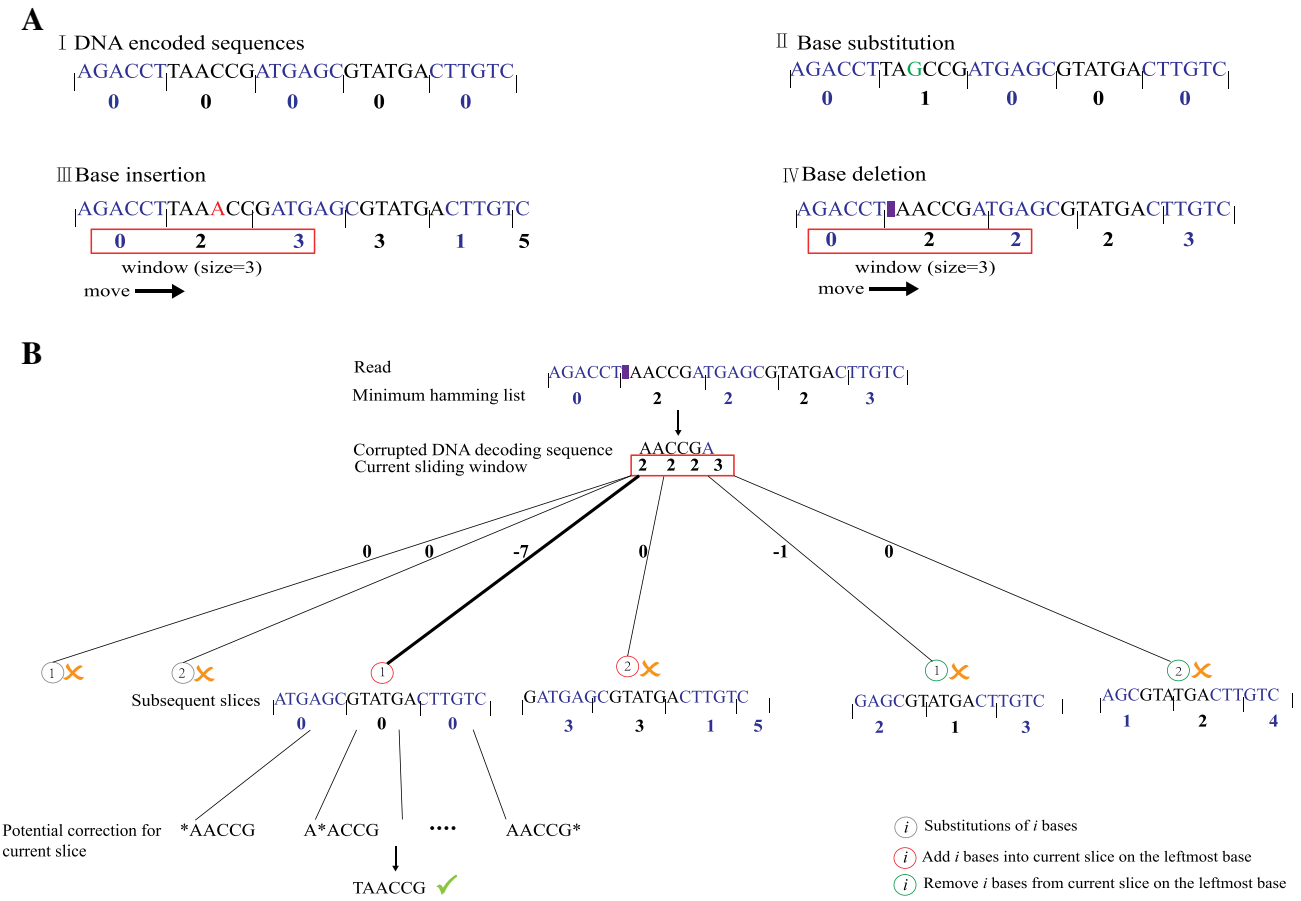

Fig. 2 Error correction of DNA sequence. **A** Minimal Hamming distance list of the original sequence and one-base error sequence. **B** Error correction for a base error

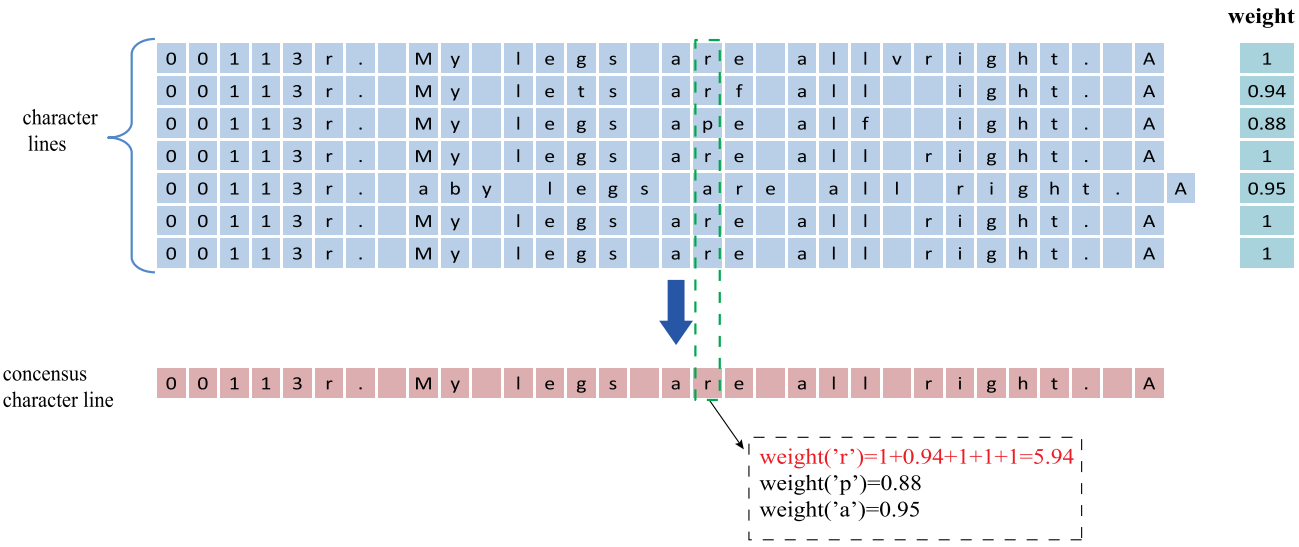

Fig. 3 An example of consensus character line generated through majority voting algorithm

their indexes and purified them. Finally, alignment the lines with the same index and output a consensus line by majority voting based on their weights as shown in Fig. 3.

Finally, the words having potential errors obtained from previous step will go through a spelling error detection and correction by the package hunspell ('https://pypi.org/project/hunspell/').

3 Results

In this paper, we use two novels "The Old Man and The Sea" and "Robert Louis Stevenson, A Record, An Estimate, A Memorial" (total about 324 KB) as English text materials which include 59,055 English words in total. Figure 4A shows the structure of the DNA sequences. The first 30nt (nucleotide, nt) encode indexes from 0 to 99,999 and the remained nucleotides are used for the data payload. The text of the two novels is finally encoded by 11,637 DNA sequences with length 210 nt.

To test the feasibility of the proposed method, we set up a series of computer simulation experiments to search a suboptimal code mapping which is not prone to form DNA sequences with secondary structure, to test the performance of the proposed error correction method in different error rates, and to analyze the robustness of the codes in Table 1. Finally, we give a brief comparative analysis of our method with other methods. All numerical experiments are done on an Ubuntu computer with six Core 3.2 GHz processors (Intel(R) Core(TM) i7-8700 CPU at 3.20 GHz) and 8 GB of physical memory.

3.1 The Optimization of the Code Assignment

A DNA sequence is prone to form secondary structure if it contains some subsequences which are reverse complementary. For example, 5′-ACTCTCCATGGACAA-3' include two subsequences 5′-CTCC-3′ and 5′-GGAC-3′ which are complementary to form a partial double strand. In DNA storage, the secondary structure may be harmful in DNA synthesis, PCR and sequencing. Therefore, it is important to encode information with DNA sequences which are not prone to form such structures. In works [15, 35], sequence randomization is used to reduce the possibility of such scenario.

For text files, both the words and sentences are organized in some ordered way. We apply a random searching algorithm to find a suboptimal DNA code mapping which is not prone to produce secondary structure in DNA sequences. We define a sequence is prone to form secondary structure if it has reverse complementary subsequences with length 10nt at least. In the simulation experiment, the two text files are encoded by DNA sequences with length from 126 to 234nt. All the results are obtained from 39,000 random mapping experiments.

Figure 4B shows the distribution of the percentage of secondary structural sequences for all the code mapping. There are a few code mappings which tend to generate very few secondary structural sequences. The code mapping in Table 1 is the one with the lowest percentage (about 3.06%). Therefore, the code book strategy can control the effect of the secondary structure through optimizing the code mapping. Figure 4C further shows that the average percentage of the secondary by the code mapping in Table 1 which increases as the length of DNA sequences increases from 126 to 234 nt. For in vitro DNA storage, long DNA sequences usually provides large logic storage density. Therefore, it is better to take a tradeoff between the reliability and storage density.

3.2 Error Correction Performance

In this section, we present the performance of the proposed error correction method at error rates from 0.01 to 0.1. The encoded DNA sequence length is 210nt (30nt for index and 180 for payload). Given a specific error rate, we random

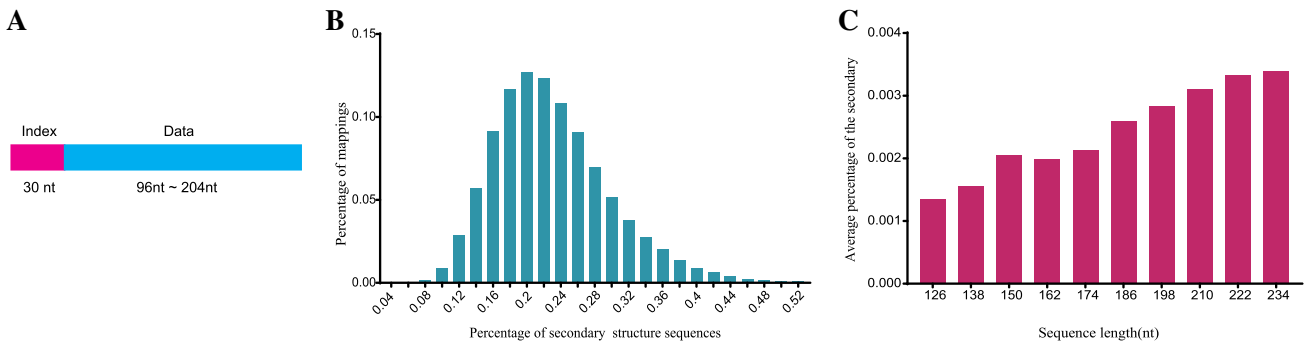

Fig. 4 Analysis of optimization designing for code table and sequence length. **A** Structure of DNA encoded sequence. **B** The distribution of percentage of secondary structure for random mappings. **C** The average rate of the secondary at different sequence length

select some positions in a sequence to add the three kinds of errors. The substitution, insertion and deletion errors are assumed to be equally distributed. The experiments are repeated 1,000 times under each pair of coverage and error rate.

Figure 5A shows the length distribution of reads at error rate 0.01–0.04, which looks like normal distributions as the three kinds of errors are assumed to be equal. The percentage of sequences in correct length decreases from 35 to 16% as error rate increases. Therefore, about 60–80% reads have incorrect length. If the three kinds of errors are not equal, especially the insertion and deletion, then the variation in read length will be more biased.

Figure 5B shows the average recovery accuracy at coverage 20–50. Given a specific coverage, the recovery accuracy decreases as the error rate increases. However, it increases as the coverage increases at a specific error rate. As the error rate gets as low as 0.01–0.02, the proposed methods can recover even more than 99% errors at coverage 20. When the error rate is about 0.05, the proposed method can still recover more 98% errors at coverage 50. As the error rate is very high at 0.1, the recovery accuracy drops dramatically to about 50–60%. The main reason may be that our code book can only correct most of the one-base error. When the error is more than one, we only substitute it with a possible legal code so that the current insertions or deletions may not influence the subsequent codes. Therefore, the error degree at 0.1 may be beyond the code capability in the first-level correction.

Figure 5C further shows the average proportion of the right corrections accomplished by the three levels. Obviously, the first-level correction plays a dominate role and the amount of errors corrected by it increases as the error rate increases. The error correction ability of Table 1 determines the final performance of the proposed method. As the error rate increases, the deterioration of the code corrections will

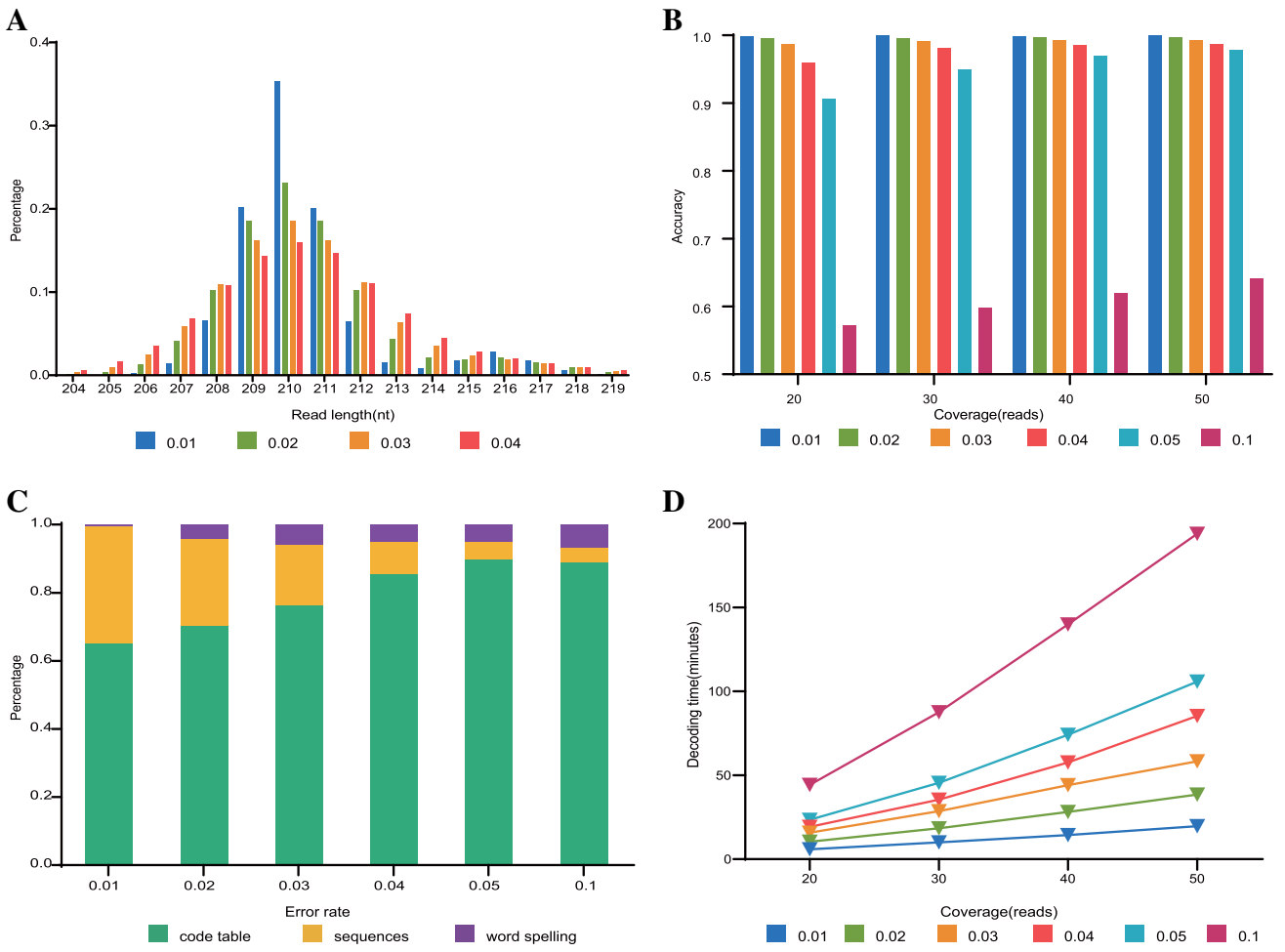

Fig. 5 Performance of proposed error correction. **A** Length distribution of reads at error rate 0.01–0.04. **B** Average recovery accuracy at different coverage and error rates. **C** Average proportion of the right corrections accomplished by the three levels at different error rates. **D** Average decoding time of the proposed method at different coverage

reduce the correction ability of the multiple alignments in the second level. Then, the proportion of the second level gradually decreases with the error rate increasing. Finally, the spelling check in the third level always helps to correct a small portion of the errors in word level.

Figure 5D shows average decoding time of the proposed method at coverage 20–50. The time need to recover the original files is linearly proportional to the coverage or error rates. In our experiment, the first-level decoding time is about 10 min and the third level takes only about 1 min. Clustering the text lines and purifying them in the second level cost most of the processing time.

3.3 The Analysis of the Code Robustness

The simple error case is that there are only substitution errors. If a code undergoes more than 1 random substitutions, it is hard to determine which legal code it comes from as the minimal Hamming distance between codes is 3. Given the substitution error as 2%, the probability of a legal code c mutated with at least 2 bases is about 0.6%. This means the probability that a code cannot be correctly distinguished is about 0.6% under substitution errors. However, when the substitution error is 10%, the probability of a legal code c mutated with at least 2 bases will increase to about 11%. Considering the complexity of insertion and deletion, this is the main reason leading to the drastic decline of accuracy.

On the other hand, a code may be mutated to other code or more similar with that code instead of itself. For example, the Hamming distance between codes 'ATCACG' and 'ATGAGC' is 3. If the last two base 'CG' in 'ATCACG' is mutated into 'CG', then it will be corrected with the second code 'ATGAGC'. If it is assumed that a base is equally mutated with probability 25%, then 'ATCACG' may be treated as 'ATGAGC' by mistake with probability 1.4%.

Considering the composite errors with substitution, insertion and deletion, it is possible that one code may be mutated to other codes through less composite mutations. For example, code 'TAAGGC' may become 'TAGGCT' with a deletion of 'A' and an insertion of 'T'. If a deletion is determined to happen in 'TAGGC', then there are two selections. One is a deletion 'A' in the middle while the other is a deletion 'T' in the end. In Table 1, the average minimal edit distance between one code and the other codes is 3.85. Figure 6 shows the distribution of the minimal edit distance between codes. There are 16 pair of codes whose edit distance is 2 among the 435 code pairs. Given the 30 codes are appeared equally, there is an about 3.6% error correction which may be incorrect based on the minimal edit distance 1.

3.4 Comparison with Other Methods

Table 2 shows the error correction strategy, indels correction, logical density, coverage, error rate, accuracy, and storage model of different methods. Compared with the text storage works in [33, 34], our proposed method can correct the three kinds of errors including indels, while the other two have no such imbedded capability, which limited their application only feasible in very low noise situations ($\leq 2\%$). The proposed method can recover more 90% data even at error rate 5% with 20X coverage while they can only deal with the help of high sequencing coverage. It is about 2700X in [33] which is not feasible for future big data storage.

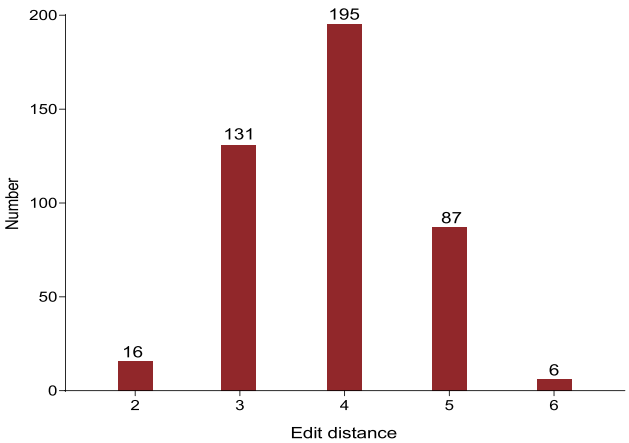

Fig. 6 The number of edit distance for 435 code pairs

Table 2 Comparison with other DNA storage methods

Study	Error correction strategy	Indels correction	Logical density	Coverage (x)	Error rate	Accuracy	Storage model
Xue et al. [30]	Levenshtein code	Yes	1.32	NA	1%	0.9	In vitro
Press et al. [31]	HEDGES	Yes	1	50	3%	1	In vitro
Song et al. [32]	de Brujin graph	Yes	1.3	60	10%	0.92	In vitro
Zhong et al. [33]	No	No	1.46	2740	$\leq 2\%$	0.998	In vitro
Lee et al. [34]	No	No	1.86	NA	$\leq 2\%$	1	In vivo
This work	Three level error correction	Yes	1.06	20	5%	0.905	In vitro

NA means not available

Compared with the three general DNA storage in [30–32], our method is more powerful than that in [31] which can correct only one-bit insertion or deletion. At error rate 3% and 50×coverage, our method and the one in [32] are almost equivalent as it can achieve more than 99% recovery accuracy. Concerning error rate 10%, the method in [32] can recover 92% information by coverage 60X, while the performance of our method drops sharply at such error rate. One drawback of the method in [32] is that it is not feasible at low coverage situations because the construction of a weighted de Brujin graph needs sufficient reads, especially when the error rate is more than 10%. However, the proposed method may have the potential to deal with large error situations at relative low coverage if we can improve the error correction ability by some more efficient algorithm in the first level. Finally, we should point out that the error correction ability of the codes in Table 1 makes the logical density of the proposed method is lower than that in [30, 32–34].

4 Conclusions

In this work, a systematic method to store English text files by DNA sequences is proposed. The main idea is to encode the common text characters with robust DNA codes and correct the various errors in DNA storage process in three levels. The proposed method has the following four advantages. First, it provides a simple code method to keep a uniform 50% GC content in all encoded sequences and avoid the homopolymer of length more than 2. The wet lab experimental results in [35] showed that these constraints may lead to about 2% read increase with corrected length and improve the decoding efficiency. Second, the undesired secondary structures in DNA sequences may be reduced and controlled through optimizing the code mapping. To our knowledge in DNA storage, this issue is considered for the first time in this paper. We believe the storage friendly encoding could remedy some degree of the "hardware" deficiencies in DNA storage. Third, the robust codes provide us the ability to correct most of the one-base insertion and deletion errors. Furthermore, if the insertions and deletions are not corrected, their downstream effects can be cut off. As shown in Fig. 5C, this step actually corrects most of the one-base errors and realign the reads to the correct length, which lays an essential foundation for the following multiple sequence alignments. Simulation results demonstrate that the proposed methods can correct more 98% errors when error rate is less than or equal to 0.05. Thus, it is more powerful and thus adaptable to the complicated DNA storage applications. Last but not least, the proposed method breaks the errors into three levels and corrects them in the corresponding level at a relatively simple way, which is actually an implementation of the famous divide-and-conquer strategy. It is worth noting that this idea can be applied to other storage applications which have some kinds of ordered structures, such as images. Finally, our future work will be focused on studying new strategy to improve the error correction ability in the first level so that it may be applied to larger error situations, such as 10 ~ 15% (Fig. 6).

Funding This work was supported by the National Natural Science Foundation of China (Grant nos. 62072128, 61876047 and 62002079).

Availability of Data and Material The data that support the findings of this study are available from the corresponding authors upon reasonable request.

Code Availability Code is available from the corresponding authors upon reasonable request.

Declarations

Conflict of Interest There is no conflict of interest.

Ethical Approval Not applicable.

Consent to Participate Not applicable.

Consent for Publication Not applicable.

References

1. Panda D, Molla KA, Baig MJ, Swain A, Behera D, Dash M (2018) DNA as a digital information storage device: hope or hype? 3 Biotech 8:239. https://doi.org/10.1007/s13205-018-1246-7
2. Williams ED, Ayres RU, Heller M (2002) The 1.7 kilogram microchip: energy and material use in the production of semiconductor devices. Environ Sci Technol 36:5504–5510. https://doi.org/10.1021/es049890z
3. Goda K, Kitsuregawa M (2012) The history of storage systems. Proc IEEE 100:1433–1440. https://doi.org/10.1109/JPROC.2012.2189787
4. Reinsel D, Gantz J, and R. J (2018) The digital of the world from edge to core[EM/OL]. http://book.itep.ru/depository/dig_economy/idc-seagate-dataage-whitepaper.pdf.
5. Bonnet J, Colotte M, Coudy D, Couallier V, Portier J, Morin B, Tuffet S (2010) Chain and conformation stability of solid-state DNA: implications for room temperature storage. Nucleic Acids Res 38:1531–1546. https://doi.org/10.1093/nar/gkp1060
6. Qian L, Ouyang Q, Ping Z, Sun F, Dong Y (2020) DNA storage: research landscape and future prospects. Natl Sci Rev 7:1092–1107. https://doi.org/10.1093/nsr/nwaa007
7. Ceze L, Nivala J, Strauss K (2019) Molecular digital data storage using DNA. Nat Rev Genet 20:456–466. https://doi.org/10.1038/s41576-019-0125-3
8. Heckel R, Mikutis G, Grass RN (2018) A characterization of the DNA data storage channel. Sci Rep. https://doi.org/10.1038/s41598-019-45832-6
9. Cretu Stancu M, van Roosmalen MJ, Renkens I, Nieboer MM, Middelkamp S, de Ligt J, Pregno G, Giachino D, Mandrile G, Espejo Valle-Inclan J, Korzelius J, de Bruijn E, Cuppen E, Talkowski ME, Marschall T, de Ridder J, Kloosterman WP (2017) Mapping and phasing of structural variation in patient genomes

using nanopore sequencing. Nat Commun 8:1326. https://doi.org/10.1038/s41467-017-01343-4
10. Kumar UK, Umashankar BS (2007) Improved Hamming Code for Error Detection and Correction. Proc Int Symp Wirel Pervasive Comput. https://doi.org/10.1109/ISWPC.2007.342654
11. Takahashi CN, Nguyen BH, Strauss K, Ceze L (2019) Demonstration of end-to-end automation of DNA data storage. Sci Rep 9:4998. https://doi.org/10.1038/s41598-019-41228-8
12. Blawat M, Gaedke K, Huetter I, Chen X-M, Turczyk B, Inverso S, Pruitt B, Church G (2016) Forward error correction for DNA data storage. Proced Comput Sci 80:1011–1022. https://doi.org/10.1016/j.procs.2016.05.398
13. Chen WG, Wang LX, Han MZ, Han CC, Li BZ (2020) Sequencing barcode construction and identification methods based on block error-correction codes. Sci China Life Sci 63:1580–1592. https://doi.org/10.1007/s11427-019-1651-3
14. Meiser LC, Antkowiak PL, Koch J, Chen WD, Kohll AX, Stark WJ, Heckel R, Grass RN (2020) Reading and writing digital data in DNA. Nat Protoc 15:86–101. https://doi.org/10.1038/s41596-019-0244-5
15. Antkowiak PL, Lietard J, Darestani MZ, Somoza MM, Stark WJ, Heckel R, Grass RN (2020) Low cost DNA data storage using photolithographic synthesis and advanced information reconstruction and error correction. Nat Commun 11:5345. https://doi.org/10.1038/s41467-020-19148-3
16. Grass RN, Heckel R, Puddu M, Paunescu D, Stark WJ (2015) Robust chemical preservation of digital information on DNA in silica with error-correcting codes. Angew Chem Int Ed Engl 54:2552–2555. https://doi.org/10.1002/anie.201411378
17. Chen W, Han M, Zhou J, Ge Q, Wang P, Zhang X, Zhu S, Song L, Yuan Y (2021) An artificial chromosome for data storage. Natl Sci Rev. https://doi.org/10.1093/nsr/nwab028
18. Deng L, Wang YX, Noor-A-Rahim M, Guan YL, Shi ZP, Gunawan E, Poh CL (2019) Optimized code design for constrained DNA data storage with asymmetric errors. IEEE Access 7:84107–84121. https://doi.org/10.1109/ACCESS.2019.2924827
19. Lu XZ, Jeong J, Kim JW, No JS, Park H, No A, Kim S (2020) Error rate-based log-likelihood ratio processing for low-density parity-check codes in DNA storage. Ieee Access 8:162892–162902. https://doi.org/10.1109/ACCESS.2020.3021700
20. Hou HX, Shum KW, Chen MH, Li H (2016) BASIC codes: low-complexity regenerating codes for distributed storage systems. IEEE Trans Inf Theory 62:3053–3069. https://doi.org/10.1109/TIT.2016.2553670
21. Organick L, Ang SD, Chen YJ, Lopez R, Yekhanin S, Makarychev K, Racz MZ, Kamath G, Gopalan P, Nguyen B, Takahashi CN, Newman S, Parker HY, Rashtchian C, Stewart K, Gupta G, Carlson R, Mulligan J, Carmean D, Seelig G, Ceze L, Strauss K (2018) Random access in large-scale DNA data storage. Nat Biotechnol 36:242–248. https://doi.org/10.1038/nbt.4079
22. Erlich Y, Zielinski D (2017) DNA fountain enables a robust and efficient storage architecture. Science 355:950–954. https://doi.org/10.1126/science.aaj2038
23. Anavy L, Vaknin I, Atar O, Amit R, Yakhini Z (2019) Data storage in DNA with fewer synthesis cycles using composite DNA letters. Nat Biotechnol 37:1229–1236. https://doi.org/10.1038/s41587-019-0240-x
24. Ross MG, Russ C, Costello M, Hollinger A, Lennon NJ, Hegarty R, Nusbaum C, Jaffe DB (2013) Characterizing and measuring bias in sequence data. Genome Biol 14:R51. https://doi.org/10.1186/gb-2013-14-5-r51
25. Church GM, Gao Y, Kosuri S (2012) Next-generation digital information storage in DNA. Science 337:1628. https://doi.org/10.1126/science.1226355
26. Goldman N, Bertone P, Chen S, Dessimoz C, Leproust EM, Sipos B, Birney E (2013) Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. Nature 494:77–80. https://doi.org/10.1038/nature11875
27. Bornholt J, Lopez R, Carmean D, Ceze L, Seelig G, Strauss K (2017) A DNA-based archival storage system. ACM SIGPLAN Notices 51(4):637–649. https://doi.org/10.1145/2954679.2872397
28. Wang Y, Noor-A-Rahim M, Zhang J, Gunawan E, Guan YL, Poh CL (2019) High capacity DNA data storage with variable-length Oligonucleotides using repeat accumulate code and hybrid mapping. J Biol Eng 13:89. https://doi.org/10.1186/s13036-019-0211-2
29. Zhang SF, Peng K (2020) DNA information storage technology based on raptor code. Laser Optoelectron. https://doi.org/10.3788/LOP57.151701
30. Xue TB, Lau FCM (2020) Construction of GC-balanced DNA with deletion/insertion/mutation error correction for DNA storage system. IEEE Access 8:140972–140980. https://doi.org/10.1109/ACCESS.2020.3012688
31. Press WH, Hawkins JA, Jones SK, Schaub JM, Finkelstein IJ (2020) HEDGES error-correcting code for DNA storage corrects indels and allows sequence constraints. Proc Natl Acad Sci USA 117:18489–18496. https://doi.org/10.1073/pnas.2004821117
32. Song L, Geng F, Gong Z, Li B, Yuan Y (2020) Super-robust data storage in DNA by de Bruijn graph-based decoding. BioRxiv. https://doi.org/10.1101/2020.12.20.423642
33. Zhong Y, Qi S, Sheng F et al (2018) A new digital information storing and reading system based on synthetic DNA. Sci China Life Sci 61:733–735. https://doi.org/10.1007/s11427-017-9131-7
34. Lee UJ, Hwang S, Kim KE, Kim M (2020) DNA data storage in perl. Biotechnol Bioprocess Eng 25:607–615. https://doi.org/10.1007/s12257-020-0022-9
35. Jeong J, Park SJ, Kim JW, No JS, Jeon HH, Lee JW, No A, Kim S, Park H (2021) Cooperative sequence clustering and decoding for DNA storage system with fountain codes. Bioinformatics. https://doi.org/10.1093/bioinformatics/btab246