JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 26, Number 0, 2019
© Mary Ann Liebert, Inc.
Pp. 1–9
DOI: 10.1089/cmb.2018.0173

Research Article

Encoding and Decoding DNA Sequences by Integer Chaos Game Representation

CHANGCHUAN YIN

ABSTRACT

DNA sequences are fundamental for encoding genetic information. The genetic information may be understood not only from symbolic sequences but also from the hidden signals inside the sequences. The symbolic sequences need to be transformed into numerical sequences so the hidden signals can be revealed by signal processing techniques. All current transformation methods encode DNA sequences into numerical values of the same length. These representations have limitations in the applications of genomic signal compression, encryption, and steganography. We propose a novel integer chaos game representation (inter-CGR or iCGR) of DNA sequences and a lossless encoding method DNA sequences by the iCGR. In the iCGR method, a DNA sequence is represented by the iterated function of the nucleotides and their positions in the sequence. Then the DNA sequence can be uniquely encoded and recovered using three integers from iCGR. One integer is the sequence length and the other two integers represent the accumulated distributions of nucleotides in the sequence. The integer encoding scheme can compress a DNA sequence by 2 bits per nucleotide. The integer representation of DNA sequences provides a prospective tool for sequence analysis and operations.

Keywords: chaos game representation, compression, decoding, DNA sequences, encoding.

1. INTRODUCTION

In recent years, the next-generation sequencing (NGS) techniques have resulted in massive DNA and protein sequences. There are strong demands for efficiently analyzing these genomic sequences. A DNA sequence consists of four types of nucleotides: adenine (A), guanine (G), thymine (T), and cytosine (C). DNA sequence analysis requires conversion of a symbolic sequence to a numerical sequence so that intrinsic patterns and characters can be characterized by digital signal processing approaches (Anastassiou, 2000; Mendizabal-Ruiz et al., 2017; Yin and Yau, 2008; Yin and Wang, 2016). Numerical representations of DNA sequences are also essential to genome comparison, compression, encryption, and steganography.

An effective numerical representation must be able to capture all significant properties of the biological reality without introducing any spurious effects. Currently, the most commonly used encoding method is the Voss 4D binary indicator representations (Felsenstein et al., 1982; Voss, 1992), which has been used in protein-coding prediction, similarity analysis, and periodicity detection in genomes. However, the Voss 4D method and DNA sequence mapping are not one-to-one. In 1990, Jeffrey first proposed a numerical and graphical chaos game representation (CGR) of a DNA sequence (Jeffrey, 1990). The CGR is generated in a

Department of Mathematics, Statistics, and Computer Science, The University of Illinois at Chicago, Chicago, Illinois.

square with the four vertices for the nucleotides A, C, G, and T, respectively. In the CGR graph, the first point is placed on the halfway position between the center of the unit square and the vertex of the unit square for the first nucleotide of the DNA sequence. The successive point is generated on the halfway position between the previous CGR point and the vertex for the current nucleotide of the sequence. An important feature of the CGR is that the value of any point in CGR contains the historical information of the preceding sequence and visually displays all subsequent frequencies of a given DNA sequence. The CGR preserves all statistical properties of DNA sequences and allows investigation of both local and global patterns in DNA sequences, visually revealing previously hidden sequence structures. The CGR was then developed for k-mer counting and referred to frequency CGR, which renders a unique two-dimensional (2D) image signature for a genome sequence.

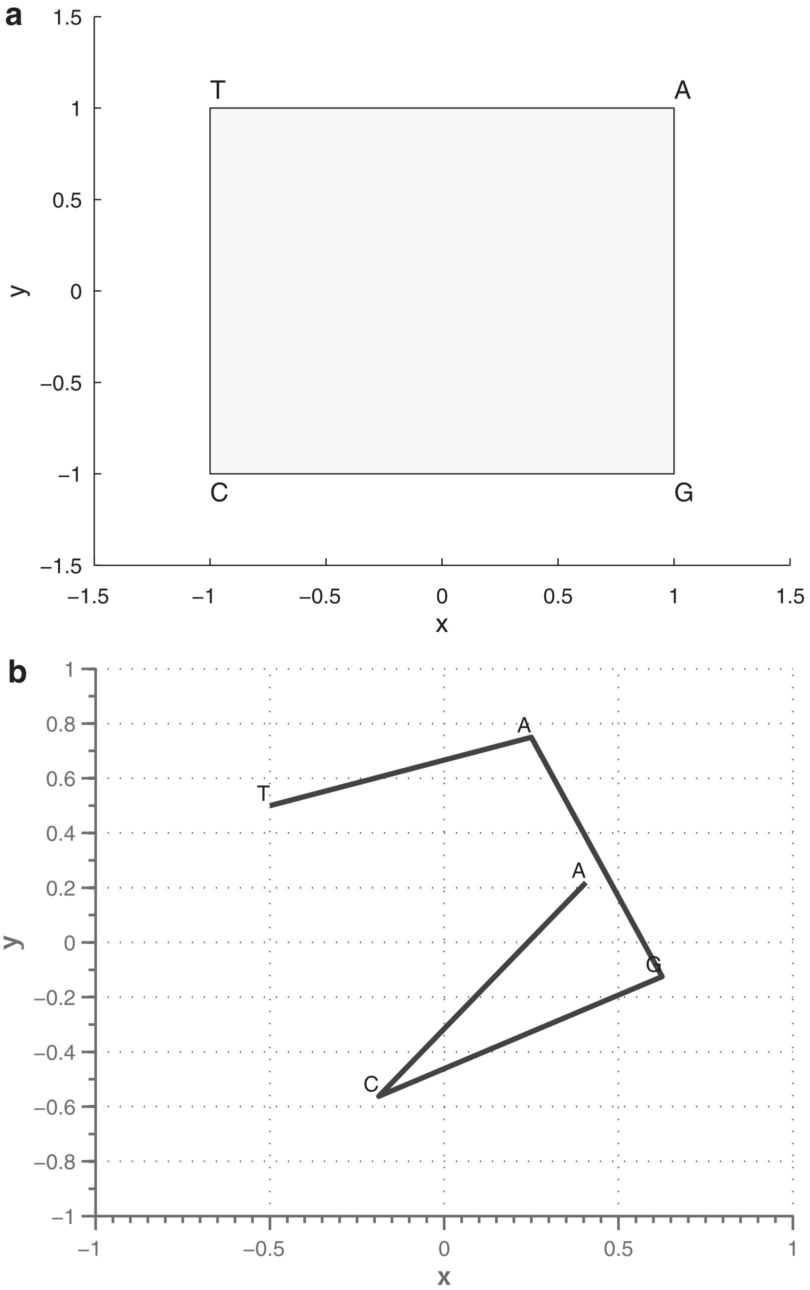

FIG. 1. (**a**) Numerical representation of four nucleotides A, T, C, and G in CGR. (**b**) The CGR graph of the DNA sequence, TAGCA. The corresponding CGR coordinates are: $T_1(-0.5000, 0.5000), A_2(0.2500, 0.7500), G_3(0.6250, -0.1250), C_4(-0.1875, -0.5625), A_5(0.4063, 0.2188)$. A, adenine; C, cytosine; CGR, chaos game representation; G, guanine; T, thymine.

Because CGR has a remarkable ability to differentiate between genetic sequences belonging to different species, it has thus been proposed as a genomic signature (Deschavanne et al., 1999; Almeida et al., 2001). Owing to the character of information preservation of CGR, it has been applied in different research domains, including similarity analysis of genomes (Stan et al., 2010; Kari et al., 2015; Joseph and Sasikumar, 2006; Hoang et al., 2016) and detection of hidden periodicity signal in genomes (Messaoudi et al., 2014; Yin and Yau, 2005; Yin et al., 2014). However, all existing numerical representation methods of DNA sequences produce a list of values of the same length of DNA sequences, and these types of representations cannot be directly used for storing, compressing, encrypting, and aligning DNA sequences.

In this article, we propose an integer chaos game representation (iCGR) of DNA sequences, in which nucleotides of DNA sequences are represented by iterated integer functions. Using iCGR, a DNA sequence can be uniquely encoded and recovered by three integers. One of the integers is the length of the DNA sequence, and the other two integers are determined by the type and positions of nucleotides in the DNA sequence. One application of the encoding is to compress DNA sequences. The result shows that 2 bits are required for storing a nucleotide symbol in integer encoding, whereas the common character representation of a nucleotide needs 8 bits. The proposed method will have wide applications in NGS analysis.

2. METHODS AND ALGORITHMS

2.1. CGR of DNA sequences

CGR is an iterative mapping and scale-independent representation for geometric representation of DNA sequences (Jeffrey, 1990). The CGR space can be viewed as a continuous reference system, where all possible sequences of any length occupy a unique position. The position is produced by the four possible nucleotides, which are treated as vertices of a unit binary square since a DNA sequence can be treated formally as a string of the four letters A, C, G, and T. In this study, we redesign the CGR corners of four nucleotides so the relationship between two nucleotides can be reflected by the distances of the CGR corners. The CGR vertices are assigned to the four nucleotides as $A = (1, 1)$, $T = (-1, 1)$, $C = (-1, -1)$, and $G = (1, -1)$ (Fig. 1a). The CGR coordinates are calculated iteratively by moving a pointer to half the distance between the previous position and the current binary representation (Algorithm 1). For example, if the next nucleotide on the DNA sequence is G, then a point is plotted halfway between the previous point and the G corner. The resulting graphic is planar, thus we call it as classical CGR representation. A CGR example of a short DNA sequence, TAGCA, is illustrated in Figure 1b. The CGR representation of human mitochondrial genome is shown in Figure 2. The CGR of human mitochondrial genome reveals the fractal patterns within the genome.

Algorithm 1: Algorithm for Computing the CGR of a DNA Sequence

Input: A DNA sequence S of length n

Output: List of 2D coordinates of the DNA sequence,
$p_i(x, y), i = 1, 2, \cdots, n$

Step:

1. Create a square with four corners
$A = (1, 1)$, $T = (-1, 1)$, $C = (-1, -1)$, and $G = (1, -1)$, representing the four nucleotides A, T, C, and G, respectively. These four points are denoted as CGR corners $\alpha \in \{A, T, C, G\}$.
2. Initialize the first CGR coordinate based on the first nucleotide, $S(1)$, of the DNA sequence.

$$\begin{aligned} p_{1,x} &= \tfrac{1}{2}\alpha_{1,x} \\ p_{1,y} &= \tfrac{1}{2}\alpha_{1,y} \\ \alpha_1 &= S(1), \alpha_1 \in \{A, T, C, G\} \end{aligned} \tag{1}$$

3. Compute the CGR coordinate of current nucleotide, $S(i)$, of the DNA sequence as the midpoint of previous coordinate and the CGR corner for this nucleotide.

$$\begin{aligned} p_{i,x} &= \tfrac{1}{2}\left(p_{i-1,x} + \alpha_{i,x}\right) \\ p_{i,y} &= \tfrac{1}{2}\left(p_{i-1,y} + \alpha_{i,y}\right) \\ \alpha_i &= S(i), \alpha_i \in \{A, T, C, G\} \\ i &= 2, 3, \ldots, n \end{aligned} \tag{2}$$

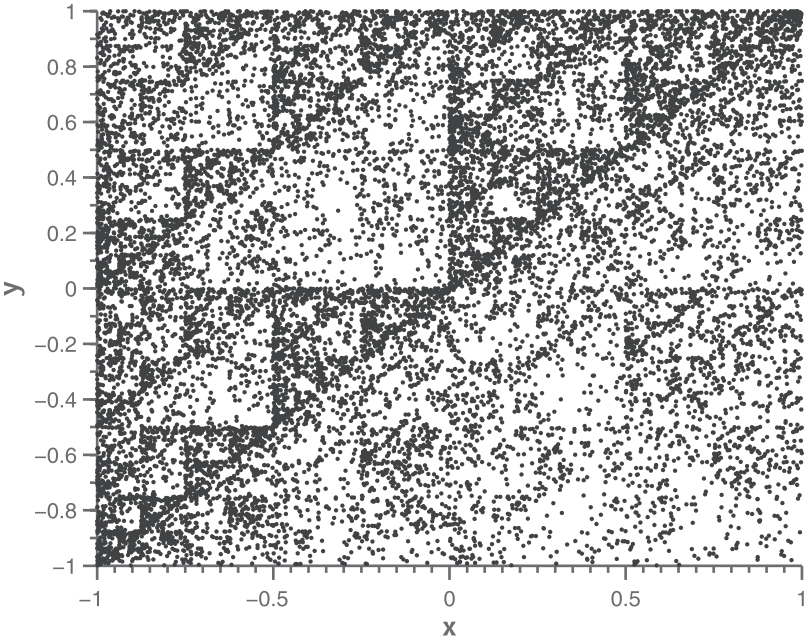

FIG. 2. The CGR graph of human mitochondrial genome (GenBank access no.: D38116).

2.2. *Encoding DNA sequences by iCGR*

From the definition of CGR, we notice that the CGR coordinates of current nucleotide is determined by the CGR coordinates of the preceding nucleotide and the fixed CGR corner coordinates of the current nucleotide. According to this recursive relationship, we may get the final CGR coordinates of a DNA sequence. The CGR theorem suggests that the final coordinates contain the full DNA sequence information (Jeffrey, 1990). However, due to the floating-point errors in the computation of CGR, DNA sequences cannot be fully recovered by the final CGR coordinates. We hereto solve this lossless encoding and decoding problems that occur in the classical CGR representation.

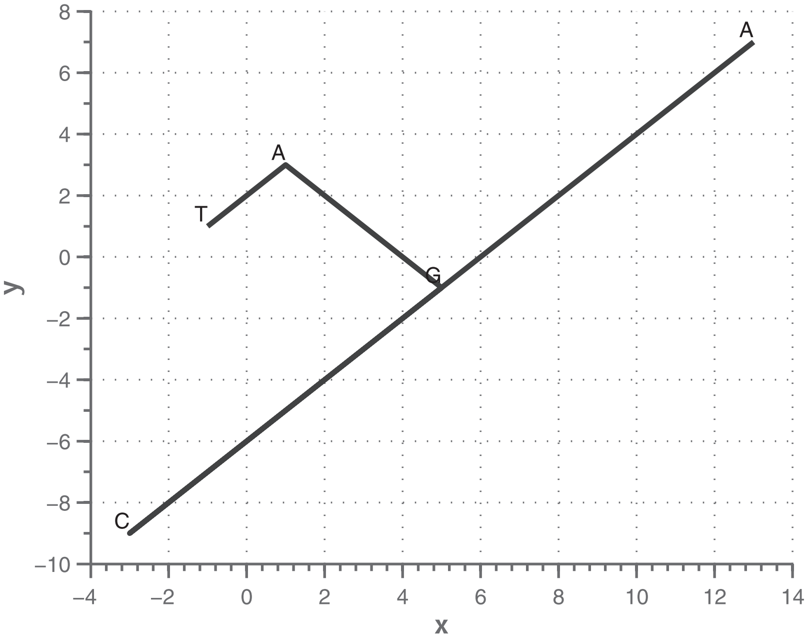

FIG. 3. iCGR encoding the DNA sequence, TAGCA. The corresponding iCGR coordinates are: $T_1(-1, 1)$, $A_2(1, 3)$, $G_3(5, -1)$, $C_4(-3, -9)$, $A_5(13, 7)$. iCGR, integer chaos game representation.

To address the floating-point errors when encoding DNA sequences in the original CGR scheme, here we redefine a new CGR coordinate relationship as in Equations (3) and (4). Instead of taking the midpoint of the preceding position and the current CGR corner as in original CGR (Jeffrey, 1990), the current position of the new CGR schema is the sum of the preceding coordinate and exponential of two of the CGR corner [Eq. (4)]. As an example, the iCGR of a short DNA sequence, TAGCA, is illustrated in Figure 3. It should be noted that the proposed CGR mapping of DNA sequences is different from the original CGR. In the proposed CGR mapping, the coordinates of DNA sequences are integers and can extend all the 2D space, whereas regular CGR coordinates are float numbers and are limited to the unit square. Thus, we may consider that the proposed iCGR is an open mapping, and the original CGR is a closed mapping. Although the original CGR coordinates are determined by the DNA sequences from theoretical analysis, due to floating-point errors, the original CGR cannot recover a DNA sequence when the length of the sequence is longer than 32 bp. Our proposed CGR mapping results in integer coordinates without the floating-point errors; therefore, the iCGR can recover a full DNA sequence when the length of the sequence is <1024 bp. This is the significant advantage of our proposed iCGR mapping.

We propose here an iCGR encoding algorithm for a DNA sequence. Using this algorithm, a DNA sequence can be uniquely represented by three numbers: the length of the sequence and the two integers of the final CGR coordinate of the DNA sequence. These integers contain all the DNA sequence information and can recover the sequence reversely. Encoding a DNA sequence into three integers by iCGR is as follows [Eq. (3)] and (Algorithm 2). We first initialize the CGR coordinate at the first position of a DNA sequence using the CGR corner coordinate, and then the following CGR coordinate is computed based on the preceding coordinate and the nucleotide at this position.

$$\begin{array}{l} p_{1,x} = \alpha_{1,x} \\ p_{1,y} = \alpha_{1,y} \\ \alpha_1 = S(1), \alpha_1 \in \{A, T, C, G\} \end{array}. \tag{3}$$

Then we can get all the current iCGR coordinates at position i based on the preceding coordinate at position $i-1$ and the nucleotide at position i.

$$\begin{array}{l} p_{i,x} = p_{i-1,x} + 2^{i-1}\alpha_{i,x} \\ p_{i,y} = p_{i-1,y} + 2^{i-1}\alpha_{i,y} \\ a_i = S(i), a_i \in \{A, T, C, G\}. \\ i = 2, \ldots, n \end{array} \tag{4}$$

From the recursive relation [Eq. (4)], we may prove that the iCGR coordinates are the sum of the product of position exponents and nucleotide types [Eq. (5)]. When a DNA sequence of length n is finally encoded by the three integer numbers $(n, p_{n,x}, p_{n,x})$ (tri-integers) through the iCGR, these three integers reflect the accumulative distribution of nucleotides along the sequence. Because the final encoding tri-integers hold all the sequence information, we propose to use the encoding tri-integers as the signature of a DNA sequence.

$$\begin{array}{l} p_{n,x} = \sum_{i=1}^{n} 2^{i-1}\alpha_{i,x} \\ p_{n,y} = \sum_{i=1}^{n} 2^{i-1}\alpha_{i,y} \\ \alpha_i = S(i) \\ i = 1, 2, \cdots, n \end{array}. \tag{5}$$

2.3. *Integer decoding DNA sequences*

We can prove that the sign of p_i is the same as that of nucleotide α_i. Therefore, the iCGR coordinates can be used to determine the corresponding nucleotide types. The nucleotide at position i can be determined based on the iCGR coordinate at this position $p(x_i, y_i)$ using the properties as in Equations (6) and (7). This can lead to the full recovery of original sequence from the final tri-integers (n, X, Y).

$$\alpha_i = \begin{cases} A, \exists \left(p_{i,x} > 0 \ \& \ p_{i,y} > 0\right) \\ \mathrm{T}, \exists \left(p_{i,x} < 0 \ \& \ p_{i,y} > 0\right) \\ C, \exists \left(p_{i,x} < 0 \ \& \ p_{i,y} < 0\right) \\ G, \exists \left(p_{i,x} > 0 \ \& \ p_{i,y} < 0\right) \\ \end{cases} . \tag{6}$$

We have the following theorem on integer encoding DNA sequences.

Theorem 2.1. *When a DNA sequence of length n is encoded by the CGR coordinates* $(X, \ Y)$, *the sequence information can be fully recovered from tri-integers* $(n, \ X, \ Y)$.

After a DNA sequence of length n is encoded recursively by the iCGR methd, then the sequence can be represented by the encoding integers $(n, \ X, \ Y)$ of the last step of iCGR encoding. In the tri-integers (n, X, Y) that a DNA sequence of length n is encoded by the iCGR method, X is the x coordinate of iCGR at position n and Y is the y coordinate of iCGR at position n. Thus we can decode and recover the original DNA sequence from these tri-integers (n, X, Y). The first step in the decoding process is to determine the nucleotide α_n at the last position n. From Equation (6), we may determine the nucleotide α_n. For example, if $X = -19$, $Y = -25$, then the nucleotide at n is A. Then we can recover the second last coordinate $\left(p_{n-1,x}, p_{n-1,y}\right)$ from the last coordinate (x_n, y_n) and the vertex coordinate of last nucleotide α_n [Eq. (7)]. After we get $\left(p_{n-1,x}, p_{n-1,y}\right)$, we may determine the nucleotide α_{n-1} at position $n-1$ from $\left(p_{n-1,x}, p_{n-1,y}\right)$ by Equation (6). Using this method, all the nucleotides at positions $i = n-1, \ n-2, \ \ldots, 1$ can be recursively determined.

By this encoding method, we can see that any DNA sequences that end with nucleotide A are encoded by two large integers in Quadrant I, those end with nucleotide T are in Quadrant II, those end with nucleotide C are in Quadrant III, and those end with nucleotide G are in Quadrant IV. From the locations of the sequences, we can determine the type of the last nucleotide of the sequences.

After a DNA sequence is encoded into tri-integers by the iCGR scheme, the sequence can be fully recovered from the tri-integers. To recover the DNA sequence from encoded tri-integers $(n, \ X, \ Y)$, we can first determine the last nucleotide according to Equation (6). Since the CGR corner coordinate α_n is known, we can use the following induction to obtain the CGR coordinate of $n-1$ position, and then determine the nucleotide by Equation (7). Using the iteration process, all the nucleotides at all the positions, $n-1, n-2, \ \ldots.1$, can be determined (Algorithm 3).

The tri-integers can also detect single nucleotide mutation in a DNA sequence. For example, the mutation from A to T in a DNA sequence, of which the wild-type sequence is represented by tri-integer (n, X, Y), the mutation generates new tri-integers $(n, \ X_m, \ Y_m)$. The difference of (n, X, Y) and (n, X_m, Y_m) can determine the single nucleotide mutation.

It is noted that the encoding scheme can detect an error if the given tri-integers are not for a DNA sequence. In each step of decoding, the value of each nucleotide can be recovered. If the values are not 1/-1 pairs, then the given tri-integers are not for a DNA sequence. Therefore, when DNA sequences are encoded by the proposed iCGR, if there is an error during data storage and transfer, the iCGR encoding and decoding method can detect this error at the location.

$$\begin{aligned} &p_{i-1,x} = p_{i,x} - 2^i \alpha_{i,x} \\ &p_{i-1,y} = p_{i,y} - 2^i \alpha_{i,y} \\ &i = 2, \ \ldots, n \end{aligned} . \tag{7}$$

Algorithm 2: Encoding a DNA Sequence by Tri-Integers (n, X, Y)

```
Input: A DNA sequence $S$ of length $n$
Output: Tri-integers $(n, X, Y)$ representing the DNA sequence
Step:
  1. Get the nucleotide coordinates $\alpha_{1,x}, \alpha_{1,y}$ at position 1 based on Equation (3).
  2. Compute the x-coordinate at position $i$ from that at position $i-1$: $p_{i,x} = p_{i-1,x} + 2^{i-1}\alpha_{i,x}$.
  3. Compute the x-coordinate at position $i$ from that at position $i-1$: $p_{i,y} = p_{i-1,y} + 2^{i-1}\alpha_{i,y}$.
  4. Repeat steps 2 and 3 until $i = n$.
  5. When $i = n$, return $X = p_{n,x}$, $Y = p_{n,y}$.
```

Table 1. Encoding a DNA Sequence of Length 10 by Tri-Integers (n, X, Y)

DNA	C	G	T	A	A	C	T	A	G	T
i	1	2	3	4	5	6	7	8	9	10
$\alpha_{i,x}$	–1	1	–1	1	1	–1	–1	1	1	–1
$\alpha_{i,y}$	–1	–1	1	1	1	–1	1	1	–1	1
$p_{i,x}$	–1	1	–3	5	21	–11	–75	53	309	–203
$p_{i,y}$	–1	–3	1	9	25	–7	57	185	–71	441

A, adenine; C, cytosine; G, guanine; T, thymine.

```
Algorithm 3: Decoding a DNA Sequence from Tri-Integers $(n, X, Y)$

Input: Tri-integers $(n, X, Y)$
Output: The DNA sequence that is encoded by tri-integers $(n, X, Y)$
Step:
    1. Get the nucleotide $\alpha_n$ at position n from $(n, X, Y)$ based on Equation (3).
    2. Compute the x-coordinate at position $i-1$ from that at position $i$: $p_{i-1,x} = p_{i,x} - 2^{i-1}\alpha_{i,x}$.
    3. Compute the y-coordinate at position $i-1$ from that at position $i$: $p_{i-1,y} = p_{i,y} - 2^{i-1}\alpha_{i,y}$.
    4. Get the nucleotide $\alpha_{i-1}$ at position $i-1$ from $p_{i-1,x}, p_{i-1,y}$ based on Equation (3).
    5. Repeat steps 2, 3, and 4 until $i = 1$.
    6. Return the decoded nucleotide sequence of length n.
```

3. RESULTS

In the encoding and decoding algorithms, we consider two classes of four different bases in DNA, the pyrimidines [cytosine (C) and thymine (T)], and the purines [adenine (A) and guanine (G)], and design new CGR corners. The algorithms use integer iteration so the relationship between nucleotide positions and encoding integers are one-to-one. We perform case studies for evaluating the effectiveness of the integer encoding and decoding algorithms in sequence representation and compression.

An example of encoding a short DNA sequence of length 10 bp is illustrated in Table 1. Table 1 shows the encoded $p_{i,x}$ and $p_{i,y}$ at each position i. The two large integers in the final step for $n = 10$ are –203 and 441. The DNA sequence can be encoded and stored by these integers, $(n = 10, X = -203, Y = 441)$.

An example of decoding a DNA sequence is illustrated in Table 2. Table 2 shows the encoded integers $p_{i,x}$ and $p_{i,y}$ and decoded nucleotide at each position i. The tri-integers $(n = 10, X = 659, Y = 783)$ are used for recovering the full sequence. The DNA sequence recovered by the tri-integers is ‘‘ATTGCCGTAA.’’

Another example of encoding is for encoding *Homo sapiens* globin gene (GenBank access no.: HF583935). For each position of the sequence, we generate the encoding iCGR coordinates as shown in Figure 4. The final encoding tri-integers for this gene are (171, 2050339409081302816541928568690764733194931295596027, 1119453162673286728512143679440023858905057236646823).

Using the encoding method, the average bits of the tri-integers per nucleotide is 2.0, whereas symbolic DNA sequences need 8 bits per nucleotide. For example, the total bits of the tri-integers for the *Homo sapiens* globin gene are 349 bits, each nucleotide needs 2.041 bits. Therefore, encoding DNA sequences as the tri-integers may save storage space.

Table 2. Decoding a DNA Sequence from Tri-Integers $(n = 10, X = 659, Y = 783)$

i	10	9	8	7	6	5	4	3	2	1
$p_{i,x}$	659	147	–109	19	–45	–13	3	–5	–1	1
$p_{i,y}$	783	271	15	–113	–49	–17	–1	7	3	1
Nucleotide	A	A	T	G	C	C	G	T	T	A

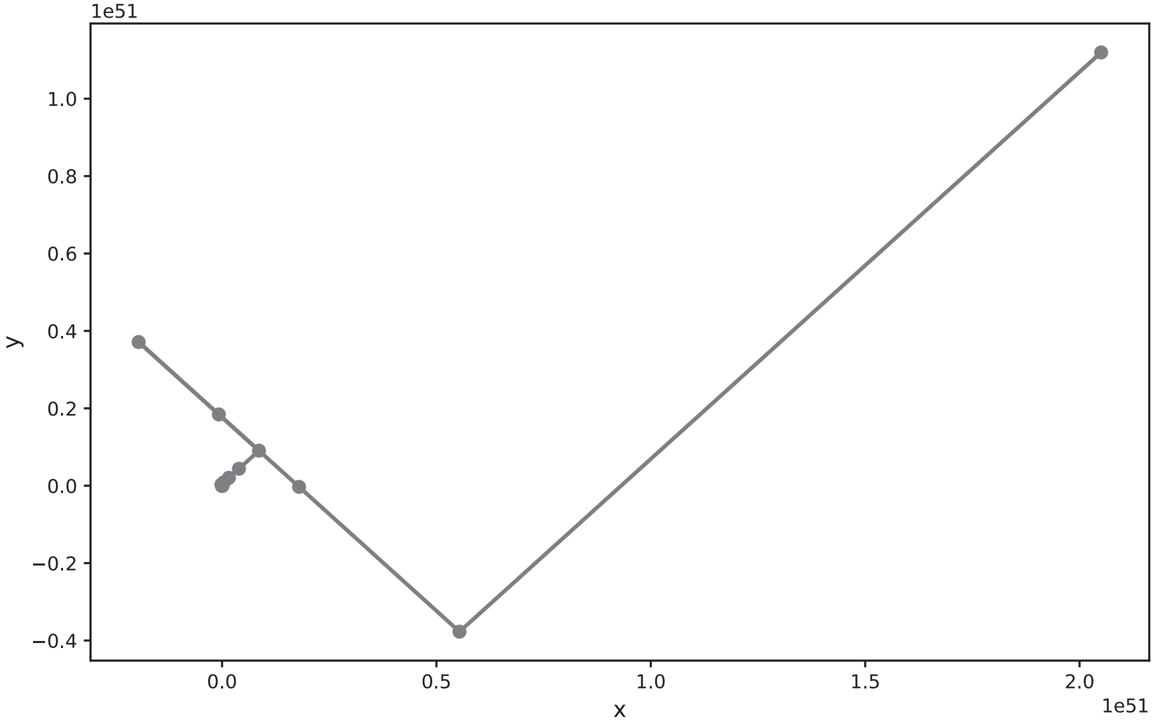

FIG. 4. Integer encoding *Homo sapiens* globin gene (GenBank access no.: HF583935).

4. CONCLUSIONS

We present a novel method for encoding a DNA sequence into three integers. Encoding a DNA sequence by iCGR produces unique tri-integers that contain all sequence information. Therefore, the tri-integers from encoding a DNA sequence can be considered as the mathematical descriptor of the sequence. The encoding method can be a promising tool for DNA sequence compressions, encryption, and steganography.

ACKNOWLEDGMENT

We are grateful to Professor Jiasong Wang at the Department of Mathematics, Nanjing University for helpful discussion.

AUTHOR DISCLOSURE STATEMENT

The author declares that no competing financial interests exist.

REFERENCES

Almeida, J.S., Carrico, J.A., Maretzek, A., et al. 2001. Analysis of genomic sequences by chaos game representation. *Bioinformatics.* 17, 429–437.

Anastassiou, D. 2000. Frequency-domain analysis of biomolecular sequences. *Bioinformatics.* 16, 1073–1081.

Deschavanne, P.J., Giron, A., Vilain, J., et al. 1999. Genomic signature: Characterization and classification of species assessed by chaos game representation of sequences. *Mol. Biol. Evol.* 16, 1391–1399.

Felsenstein, J., Sawyer, S., and Kochin, R. 1982. An efficient method for matching nucleic acid sequences. *Nucleic Acids Res.* 10, 133–139.

Hoang, T., Yin, C., and Yau, S.S.T. 2016. Numerical encoding of DNA sequences by chaos game representation with application in similarity comparison. *Genomics.* 108, 134–142.

Jeffrey, H.J. 1990. Chaos game representation of gene structure. *Nucleic Acids Res.* 18, 2163–2170.

Joseph, J., and Sasikumar, R. 2006. Chaos game representation for comparison of whole genomes. *BMC Bioinformatics.* 7, 243.

Kari, L., Hill, K.A., Sayem, A.S., et al. 2015. Mapping the space of genomic signatures. *PLoS One.* 10, e0119815.

Mendizabal-Ruiz, G., Román-Godínez, I., Torres-Ramos, S., et al. 2017. On DNA numerical representations for genomic similarity computation. *PLoS One.* 12, e0173288.

Messaoudi, I., Elloumi-Oueslati, A., and Lachiri, Z. 2014. Building specific signals from frequency chaos game and revealing periodicities using a smoothed Fourier analysis. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 11, 863–877.

Stan, C., Cristescu, C.P., et al. 2010. Similarity analysis for DNA sequences based on chaos game representation. case study: The albumin. *J. Theor. Biol.* 267, 513–518.

Voss, R.F. 1992. Evolution of long-range fractal correlations and 1/f noise in DNA base sequences. *Phys. Rev. Lett.* 68, 3805–3808.

Yin, C., and Wang, J. 2016. Periodic power spectrum with applications in detection of latent periodicities in DNA sequences. *J. Math. Biol.* 73, 1053–1079.

Yin, C., and Yau, S.S.T. 2005. A Fourier characteristic of coding sequences: Origins and a non-Fourier approximation. *J. Comput. Biol.* 12, 1153–1165.

Yin, C., and Yau, S.S.T. 2008. Numerical representation of DNA sequences based on genetic code context and its applications in periodicity analysis of genomes. In: 2008 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, Sun Valley, ID, 2008. CIBCB'08. pp. 223–227.

Yin, C., Yin, X.E., and Wang, J. 2014. A novel method for comparative analysis of DNA sequences by Ramanujan-Fourier transform. *J. Comput. Biol.* 21, 867–879.

Address correspondence to:
Dr. Changchuan Yin
Department of Mathematics, Statistics, and Computer Science
The University of Illinois at Chicago
Chicago, IL 60607-7045

E-mail: cyin1@uic.edu