Multimedia Tools and Applications
https://doi.org/10.1007/s11042-023-17046-6

An *S*-Box construction from exponentiation in finite fields and its application in RGB color image encryption

Steven T. Dougherty[1] · Joseph Klobusicky[1] · Serap Şahinkaya[2] · Deniz Ustun[3]

Received: 4 January 2023 / Revised: 24 August 2023 / Accepted: 11 September 2023

Abstract
In this study, the utilization of exponentiation in finite fields is investigated for the purpose of generating pseudo-random sequences which have a crucial role in cryptographic applications. More precisely, a novel method for generating pseudo-random sequences is proposed to construct an initial *S*-Box which is a key component in various encryption schemes. In addition to that, a shuffling algorithm that leverages the pseudo-random sequences is developed to enhance the effectiveness of the initial *S*-Box. The utilization of the proposed *S*-Box is applied to the RGB color images to showcase its performance and robustness in an image encryption scheme.

Keywords Image encryption · *S*-Box · Legendre symbol

1 Introduction

Block cipher system have a crucial role in the modern cryptography, and it is an essential building blocks for constructing strong and efficient encryption schemes. One of the important components within cryptosystem is the substitution box that known as an *S*-Box. The substitution box (*S*-Box) is used in numerous block cryptosystems; for instance, it is used in the Data Encryption Standard (DES), International Data Encryption Algorithm (IDEA), and Advanced Encryption Standard (AES). It is one of the fundamental tools to design secure block cipher systems [34, 48].

The *S*-Box provides a critical level of confusion and non-linearity for the encryption process. It operates by substituting input bit patterns with corresponding output bit patterns using a table. Its significance lies in its ability to introduce a high level of randomness into

✉ Deniz Ustun
denizustun@tarsus.edu.tr

1 University of Scranton, Scranton, PA 18518, USA

2 Department of Natural and Mathematical Sciences, Tarsus University, Faculty of Engineering, Mersin, Türkiye

3 Department of Computer Engineering, Tarsus University, Faculty of Engineering, Mersin, Türkiye

Published online: 11 October 2023

the encrypted content. An S-Box must provide a high nonlinearity to meet the requirements of secure encryption.

Recently, the chaos theory has emerged as a prominent approach for designing S-Box due to the inherent characteristics exhibited by chaotic maps. They provide an ability of intense complexity enabling the creation of S-Box that are resistant to various cryptanalysis techniques. The utilization of chaos theory in S-Box design has opened new avenues for exploring novel encryption algorithms with improved security properties and researchers have constructed S-Boxes that resist statistical attacks and exhibit excellent cryptographic properties by making use of the principles of chaos theory [3, 8, 9, 18, 22–25, 28, 30, 32, 41, 43, 44, 46, 53]. In addition to that, meta-heuristic algorithms are powerful optimization methods inspired by natural phenomena or social behavior. These algorithms can solve complex problems efficiently by exploring large solution spaces. Meta-heuristic algorithms have been widely used to design S-Boxes which play an important role in the cryptographic applications. There are numerous of research papers that are presented to literature for designing S-Boxes by using heuristic algorithms [2–6, 12, 17, 19, 23, 29, 30, 42, 46, 54]. There are also other techniques for an image encryption, for instance in [50], a method of M-ary decomposition which can be used to decompose an integer into different representations of virtual bits is presented. In this method, a non-standard image pre-encryption algorithm is proposed by using the M-ary decomposition method.

In this paper, a novel shuffling algorithm is proposed to produce pseudo-random sequences which is used to design an S-Box with high nonlinearity. The sequences that exhibit desirable pseudo-random characteristics are generated though the properties of finite fields. The S-Box which is derived from the pseudo-random sequences through a shuffling algorithm has a vital role in cryptographic systems. To measure the performance of the developed S-Box five major algebraic analyses are used: non-linearity, strict avalanche criterion (SAC), bit independence criterion (BIC), differential approximation probability (DP), and linear approximation probability (LP). In addition to that, the effectiveness of proposed S-Box is tested through extensive experimentation and analysis for RGB images that are chosen from SIPI image database (https://sipi.usc.edu/database/). The achieved results are compared with existing methods in the literature to demonstrate the superiority and efficacy of our approach.

1.1 Definitions and notations

Let n be an arbitrary positive integer and let p be a prime number such that $p-1$ is divisible by n. That is, $p \equiv 1 \pmod{n}$. Since $gcd(n, 1) = 1$, Dirichlet's Theorem gives that there are infinitely many primes of this form. This also means that p can be chosen to be arbitrarily large, namely, for any $M > 0$, there exists a prime p, where $p \equiv 1 \pmod{n}$ and $p > M$.

Recall that a primitive root of unity in $\mathbb{Z}_p$ is an element ξ such that $\xi^{p-1} = 1$ and $\xi^i \neq 1$ for $0 < i < p-1$. Of course, Fermat's Little Theorem gives that all non-zero elements satisfy $a^{p-1} = 1$. It is well known that the field $\mathbb{Z}_p$ always has a primitive root. This is easily shown by proving that the orders of every element must divide the maximal order of all elements in $\mathbb{Z}_p$ and then if M is that maximal element, they are all solutions to $x^M - 1 = 0$. Given $p-1$ solutions to this polynomial and the fact that $\mathbb{Z}_p$ is a field, we have that M must be $p-1$ and therefore there exists a primitive root, that is an element of order $p-1$. Given this result it follows that there are $\phi(p-1)$ primitive roots in $\mathbb{Z}_p$ where ϕ is the Euler totient function. Specifically, if ξ is a primitive root then ξ^a is a primitive root if and only if $gcd(a, p-1) = 1$. Given any primitive root ξ we know that every non-zero element is uniquely represented by ξ^i for some $i, 0 \leq i \leq p-2$.

In the finite field $\mathbb{F}_{p^e}$, a primitive root of unity is an element ξ, where $\xi^{p^e-1} = 1$ but $\xi^i \neq 1$ for $0 < i < p^e - 1$. Every non-zero element of a field satisfies $a^{p^e-1} = 1$ since the multiplicative group of a finite field has order $p^e - 1$. Since the multiplicative group of a field is cyclic we know that every finite field must contain a primitive root of unity. Given any finite field of order p^e and a primitive root of unity ξ, we have that ξ^a is a primitive root if and only if $gcd(a, p^e - 1) = 1$ since the multiplicative group of the field is cyclic. This gives that there are $\phi(p^e - 1)$ primitive roots in the finite field of order p^e. For example, in the field $\mathbb{F}_4 = \{0, 1, \omega, \omega^2\}$ both ω and ω^2 are primitive 3-rd roots of unity. That is there are two roots of unity and $\phi(3) = 2$. In general, we describe the field $\mathbb{F}_{p^e} = \mathbb{F}_p[x]/\langle p(x)\rangle$ where $p(x)$ is an irreducible polynomial of degree e. Then we can define the elements as polynomials with coefficients in $\mathbb{F}_p$ of degree less than or equal to $e - 1$.

Of course $\mathbb{Z}_p$ is a field as well with $e = 1$, however we write the results separately since we wish to make use of different computational tools in each case. Specifically, we have various number theoretic results in $\mathbb{Z}_p$ which we wish to exploit.

Let p be a prime. Let $\square_p = \{b \mid b = a^2 \text{ for some } b \in \mathbb{Z}_p^*\}$. It is well known that $|\square_p| = \frac{p-1}{2}$, which gives that the probability that a non-zero element is a square is $\frac{1}{2}$. We define the Legendre symbol as

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & a \in \square_p, \\ -1 & a \notin \square_p, \\ 0 & a = 0. \end{cases} \tag{1}$$

It is well known that for non-zero a, $\left(\frac{a}{p}\right) = a^{\frac{p-1}{2}} \pmod p$. Even for large p with over a 100 digits this is easily computed and in general takes only on the order of $log(\frac{p-1}{2})$ computations.

An S-Box is a 16 by 16 matrix where the elements come from the set $\{0, 1, 2, 3, \ldots, 255\}$ and every element from this set appears exactly once in the matrix. These objects are used in cryptographic applications. In general, one can think of the indices and entries as integers or one can think of the indices as a length 4 binary vectors reading the vector as a base 2 number and the entries as length 8 binary vectors reading the vector as a base 2 number.

2 Random sequences from exponentiation

We shall now show how to construct an n-sided die using exponentiation.

Theorem 2.1 *Let p be a prime, $p \equiv 1 \pmod n$. The function $\psi_n : \mathbb{Z}_p^* \to \mathbb{Z}_p^*$, defined by*

$$\psi_n(a) = a^{\frac{p-1}{n}} \tag{2}$$

takes on n different values equally often as the input runs over all possible values in $\mathbb{Z}_p^*$. These values are the n-th roots of unity in $\mathbb{Z}_p$.

Proof We know that there exists a primitive root of unity ξ. Then

$$\begin{aligned} \left((\xi^i)^{\frac{p-1}{n}}\right)^n &= (\xi^i)^{p-1} \\ &= 1. \end{aligned}$$

Therefore, all of the values are n-th roots of unity.

The n-th roots of unity are precisely

$$\xi^{j\frac{p-1}{n}},$$

where $0 \leq j \leq n-1$. We note that these n values must be distinct since ξ is a primitive root and each element in $\mathbb{Z}_p^*$ can be expressed uniquely as ξ^j.

Then $\psi_n(\xi^j) = \xi^{j\frac{p-1}{n}}$ and n divides $p-1$ so ψ_n takes on n many values equally often.□

Example 2.2 *As a simple example, consider* $p = 11$ *and* 5 *a divisor of* $11 - 1 = 10$. *The values of* ψ_5 *are:*

$$1, 4, 9, 5, 3, 3, 5, 9, 4, 1.$$

The values of ψ_2 *are:*

$$1, 10, 1, 1, 1, 10, 10, 10, 1, 10.$$

In the first case there are 5 *elements each of which appears* 2 *times and in the second there are* 2 *values each of which appears* 5 *times.*

In essence, what we have done is created an n sided die. That is, we have a deterministic function that produces n different outcomes in a manner which simulates randomness.

If n is 2 then the function ψ_2 is the Legendre function $\left(\frac{a}{p}\right)$. That is $\psi_2(a) = \left(\frac{a}{p}\right)$. It has been shown that the distribution of subsequences of fixed length of the Legendre symbol tends to uniform distribution exponentially in $\log_2(p)$, see [13]. This is what you would expect for a truly random sequence.

We can now prove a similar result for finite fields in general.

Theorem 2.3 *Let* $\mathbb{F}_{p^e}$ *be the finite field of order* p^e. *Let* n *be a positive integer that divides* $p^e - 1$. *The function* $\theta_n : \mathbb{F}_{p^e}^* \to \mathbb{F}_{p^e}$ *Defined by*

$$\theta_n(a) = a^{p^e-1}n,$$

takes on n *different values equally often.*

Proof The proof is identical to Theorem 2.1, noting that the multiplicative group of a finite field is cyclic, therefore there is an element of order $p^e - 1$. □

Given any positive integer n and a prime p, Dirichlet's theorem says that there exists infinitely many primes where $p \equiv 1 \pmod{n}$. Given such a prime we have $p^e \equiv 1 \pmod{n}$ for all $e > 0$, and so n divides $p^e - 1$. This means there are infinitely many finite fields where n divides $p^e - 1$ and that there are arbitrarily large values of p^e that are $1 \pmod{n}$.

Theorem 2.4 *Let* A *be the range of* ψ_n. *Let* $\epsilon_j \in A$. *The function*

$$E(\epsilon_1, \epsilon_2, \ldots, \epsilon_\ell) = \sum_{a=0}^{p-\ell} \prod_{j=1}^{\ell} \frac{\prod_{i \neq \epsilon_j} \psi_n(a + j - 1) - i}{\prod_{i \neq \epsilon_j} \epsilon_j - i} \tag{3}$$

gives the number of times that the sequence $\epsilon_1, \epsilon_2, \ldots, \epsilon_\ell$ *appears in the output of* ψ_n.

Proof First we notice that $\prod_{i \neq \epsilon_j} \psi_n(a + j - 1) - i = 0$ if the sequence $\psi_n(a + j - 1)$ is not exactly the sequence $\epsilon_1, \epsilon_2, \ldots, \epsilon_\ell$. Then $\prod_{i \neq \epsilon_j} \epsilon_j - i = \prod_{i \neq \epsilon_j} \psi_n(a + j - 1) - i = 0$ if it is exactly the sequence $\epsilon_1, \epsilon_2, \ldots, \epsilon_\ell$. □

3 Computational results

For $n \geq 2$ and a prime p with $p = 1 \pmod{n}$, consider the full sequence generated from exponentiation by

$$T(n, p) = (\psi_n(1), \psi_n(2), \ldots, \psi_n(p)). \tag{4}$$

As shown in Theorem 2.1, the range $R(\psi_n)$ is size n. For $l \geq 2$, we are interested in the number of occurrences of a prescribed subsequence $S = (\epsilon_1, \ldots, \epsilon_l)$ with $\epsilon_i \in R(\psi_n)$. Denote this quantity by

$$N(p, n, S) = \left| \{ j : j = 1, \ldots, p - l + 1, (T_i(n, p))_{i=j}^{j+l-1} = S) \} \right|. \tag{5}$$

A simple application of linearity of expectation shows that for a truly random sequence with elements uniformly generated in $R(\psi_n)$, the expected proportion of length l subsequences that match with S is $1/n^l$. For a single subsequence S, we give a measure of randomness which compares this expected value against the actual proportion of subsequences S found in T, denoted by

$$d(p, n, S) = \left| \frac{N(p, n, S)}{p - l + 1} - \frac{1}{n^l} \right|. \tag{6}$$

By averaging over possible prescribed subsequences, our measure of randomness for our n number generator using p is

$$D(p, n, l) = \frac{1}{n^l} \sum_{S \in (R(\psi_n))^l} d(p, n, S). \tag{7}$$

If Davenport's results for $n = 2$, see [14, 15], generalize for the exponentiation algorithm, the distribution of subsequences would tend to the uniform distribution exponentially in $\log_n(p)$. This would imply that for all $l, n \geq 2$,

$$D(p, n, l) = O(p^c), \quad c \in (-1, 0). \tag{8}$$

We have produced the following numerical experiment which gives evidence to this claim by considering sequences generated with $n = 8$. We randomly selected 100 primes p less

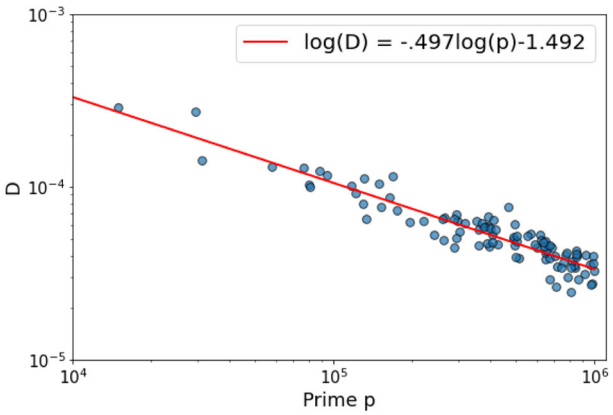


Fig. 1 Scatterplot of p versus estimates of $D(p, 8, 3)$ and the line of best fit in log-log scaling

than a million such that $p \equiv 1 \pmod 8$. For each prime p we estimated $D(p, 8, 3)$ by averaging $d(p, 8, S_i)$ over 30 randomly selected subsequences $S_1, \ldots, S_{30}$, each of length $l = |S_i| = 3$. Figure 1, shows a scatter plot of p versus estimates of $D(p, 8, 3)$ with log-log scaling, which reveals a linear trend. In the line of best fit, we have a slope of $c = -.497$, which corresponds to the value of c in (8).

4 Cryptographic applications

4.1 *S*-Boxes

We begin with a cryptographic application of Theorem 2.3.

Theorem 4.1 *Let $\mathbb{F}_{256}$ be the field of order 256. For each $\sum_{i=0}^{7} a_i \alpha^i$ in $\mathbb{F}_{256}$, let $\sigma(\sum_{i=0}^{7} a_i \alpha^i) = \sum_{i=0}^{7} a_i 2^i$. Let ξ be any of $\phi(255) = 128$ primitive roots of unity. Then let*

$$S_{i,j} = \sigma(\xi^{16i+j}), \text{ for } 0 \le i \le 7, 0 \le j \le 7.$$

Let $S_{15,15} = 0$. Then S is an S-Box.

Proof We note that (ξ^{16i+j}) gives each value of $\mathbb{F}_{256}$ exactly once, except for 1 which appears both in $S_{0,0}$ and $S_{15,15}$. Therefore, we replace the second one with 0. □

We note that this actually gives 128 S-Boxes, one for each primitive root. There is no guarantee that this S-Box has good properties in terms of its application in cryptography. This is why we describe the following algorithm to increase the nonlinearity of any S-Box.

Let S be an S-Box, where the rows and columns are indexed by 0 to 15 in their binary expansion and the elements of S are elements of the set $\{0, 1, 2, 3, \ldots, 255\}$. We indicate the element in the i-th row and j-th column of S as $S[i, j]$. We now can shuffle the S-Box using Legendre symbols as in the following algorithm.

Shuffling algorithm

1. Choose a prime p such that $p \equiv 1 \pmod{2^i}$ for i a value in $\{1, 2, 3, 4\}$. Let $a = 2^i$ for the chosen value of i.
2. Pick a $0 \le k \le p - 1$ as a starting point.
3. Compute the sequence $a_i = \psi_a(i)$.
4. Let the map μ map the a possible outputs of ψ_a to $\{0, 1, \ldots, a - 1\}$.
5. Define the sequence $b_i = \mu(a_i)$.
6. Define the sequence c_i by grouping together $\frac{16}{a}$ consecutive elements and reading this vector as a base a number.
7. Exchange the elements $S[c_{4j+1}, c_{4j+2}]$ and $S[c_{4j+3}, c_{4j+4}]$ for $j > 0$ to some arbitrary stopping point.

This switching algorithm could be done sequentially for various different values of p. One can see the Fig. 2 for the flowchart of S-Box construction by Shuffling Algorithm.

Example 4.2 *As a simple example, if we let $p = 17$ and compute $\psi_2(a)$ for $1 \le a \le 16$, we get the sequence for a_i:*

$$1, 1, -1, 1, -1, -1, -1, 1, 1, -1, -1, -1, 1, -1, 1, 1$$

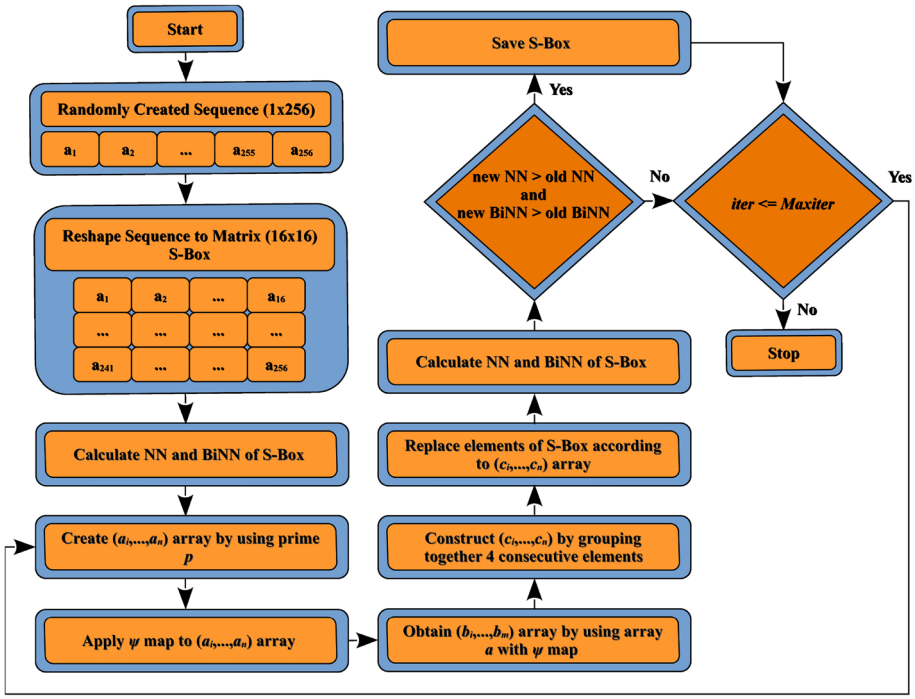


Fig. 2 A flowchart of S-Box Construction by Shuffling Algorithm

Algorithm 1 Pseudocode of S-Box construction by shuffling algorithm.

procedure CONSTRUCTIONSBOX(Maxiter)
  $S\text{-Box}_{init} \leftarrow \{s_1, s_2, ..., s_{256}\}$ ▷ Random Construction of initial S-Box
  $MaxNN \leftarrow \text{NN}(S\text{-Box}_{init})$ ▷ Calculate Nonlinearity (NN)of $S\text{-Box}_{init}$
  $MaxBiNN \leftarrow \text{BiNN}(S\text{-Box}_{init})$ ▷ Calculate Bit Independence Nonlinearity (BiNN) of $S\text{-Box}_{init}$
  $S \leftarrow S\text{-Box}_{init}$
  $S\text{-}Box_{max} \leftarrow S\text{-Box}_{init}$
  $iter \leftarrow 1$
  while $iter \leq Maxiter$ **do**
    $p \equiv 1 \mod 2^i, i \in \{1, 2, 3, 4\}$
    $k \leftarrow \lfloor (p-1) * rand(0,1) + 1 \rfloor$
    $a_i \leftarrow \mod ((k+i)^{(\frac{p-1}{4})}, p)$ ▷ a_i takes on 4 values, 1, -1 and 2 other values; I and $-I$
    $b_i \leftarrow \mu(a_i)$ ▷ $b_i \in \{0, 1, 2, 3\}$
    Define sequence c_i by group together 4 consecutive elements
    $S[c_{4j+1}, c_{4j+2}] \leftarrow S[c_{4j+3}c_{4j+4}]$ for $0 < j \leq \lfloor \frac{i}{4} \rfloor$
    $S\text{-}NN \leftarrow NN(S)$ ▷ Calculate Nonlinearity (NN)of S
    $S\text{-}BiNN \leftarrow BiNN(S)$ ▷ Calculate Bit Independence Nonlinearity (BiNN) of S
    if ($MaxNN \leq S\text{-}NN$ and $S\text{-}BiNN \leq BiNN(S)$) **then**
      $MaxNN \leftarrow S\text{-}NN$
      $MaxBiNN \leftarrow S\text{-}BiNN$
      $S\text{-}Box_{max} \leftarrow S$
    end if
    $iter \leftarrow iter + 1$
  end while
end procedure

Then the sequence b_i is:

$$0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0$$

Then the sequence c_i is $2, 14, 7, 4$. *Then we switch* $S[2, 14]$ *and* $S[7, 4]$.

Example 4.3 *Consider the prime $p = 13$. We have that* $12 \equiv 1 \pmod 4$. *Consider the values of* $a^{\frac{p-1}{4}}$.

a	1	2	3	4	5	6	7	8	9	10	11	12
$a^{\frac{p-1}{4}}$	1	8	1	12	8	8	5	5	1	12	5	12

Here we have $1, -1 = 12$, $I = 5$, $-I = 8$. *Applying the map ψ we get:*

$$0, 2, 0, 3, 2, 2, 1, 1, 0, 3, 1, 3.$$

The sequence c_i is: $2, 3, 10, 3, 3, 7$. *Then we have* $(2, 3), (10, 3), (3, 7)$. *We then switch* $S[2, 3]$ *with* $S[10, 3]$. *The examples is too small for the next pair to switch.*

4.2 One time pads

A one time pad is a sequence (generally very long) that is added to a sequence of information bits. Namely, let $s_1, s_2, \ldots s_m$ be a sequence of information from an alphabet of size n. Let $t_1, t_2, \ldots, t_m$ be the one time pad. Then the user sends $s_1 + t_1, 2_2 + t_2, \ldots, s_m + t_m$. The message is uncovered by the receiver subtracting the sequence $t_1, t_2, \ldots, t_m$ from this sequence.

The one time pad is used only once as the name indicates. It has been shown to be cryptographically secure given that it used only once. The main desire for the one time pad is that the elements of the sequence t_i appear to be random.

Assume we have an alphabet of size n. Then let p be a prime that is $1 \mod n$. Choose p so that it is greater than m. Then choose a starting place k and let $t_i = \psi_n(k + i)$. Note that $k + i$ is computed in $\mathbb{Z}_p$ so that it may wrap around.

Given a message $m_1, m_2, \ldots$ over an alphabet of size n, we encode the message by computing

$$e_i = m_i + t_i.$$

Then the message is decoded by computing

$$e_i - t_i = m_i.$$

A set of values for k and p can be agreed upon for numerous messages ahead of time, then this technique can be used efficiently to give secure information.

5 An *S*-Box constructed by the shuffling algorithm

We now present an S-Box that is constructed by the shuffling algorithm.

6 Performance analysis of the proposed *S*-Box

In this section, the proposed S-Box given in Table 1 is analyzed to measure its strength and resistance against cyber attacks. The performance of the proposed S-Box is examined through

Table 1 The proposed S-Box

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	181	126	183	81	240	80	61	232	18	4	236	156	66	169	213	33
1	220	91	163	103	247	7	115	11	73	159	226	134	121	57	173	38
2	24	212	177	130	221	95	148	194	10	84	76	171	113	210	125	200
3	254	228	162	51	168	195	179	153	63	79	56	209	23	211	123	114
4	92	160	96	218	155	204	100	214	132	140	219	106	124	205	82	5
5	161	108	136	20	67	203	244	242	164	41	215	116	52	2	229	40
6	192	19	190	146	150	110	42	111	1	253	131	27	170	118	201	246
7	69	230	112	9	47	65	58	197	135	151	86	59	117	46	174	6
8	8	54	158	147	107	99	216	199	166	109	104	0	50	62	143	105
9	49	48	193	120	227	188	75	122	83	222	45	78	139	88	185	182
10	101	71	36	43	74	234	137	189	178	85	39	249	35	72	64	175
11	138	141	167	12	44	255	245	119	26	250	248	129	13	217	224	32
12	102	28	187	37	186	68	70	208	198	98	97	89	77	152	144	55
13	30	196	202	127	154	184	17	233	22	157	231	176	29	207	237	142
14	3	34	93	53	87	235	128	149	31	243	251	241	206	191	252	238
15	133	180	16	145	21	172	165	90	25	14	225	15	60	223	94	239

five major algebraic analyses: non-linearity, strict avalanche criterion (SAC), bit independence criterion (BIC), differential approximation probability (DP), and linear approximation probability (LP).

6.1 Non-linearity

An S-Box is a nonlinear component of an encryption algorithm and an S-Box with higher nonlinearity value is more secure than that with smaller nonlinearity value. The nonlinearity of an S-Box is calculated as follows [34, 40]:

$$N(f) = 2^n(1 - 2^{-n} \max_{\omega \in GF(2^n)} \mid S_f(\omega) \mid),$$

here $S_f(\omega)$ denotes the Walsh-Hadamard transform of Boolean function f and it can be calculated by the following formula:

$$S_f(\omega) = \sum_{\omega \in GF(2^n)} (-1)^{f(x) \oplus x * \omega}.$$

The S-Box constructed by the proposed algorithm achieves the following values: 112, 110, 108, 112, 108, 110, 110, 110 and has an average value of 110. It can be seen from Table 6 that this value is larger than most of the S-Boxes generated by other latest algorithms.

6.2 Strict avalanche criterion (SAC)

The strict avalanche criterion (SAC) was proposed by Webster and Tavares in [48]. This criteria analyzes the relationship between the output bits and input bits. More explicitly, the probability of change in each of the output bits by altering a single input is expected to be

Table 2 The dependence matrix of the proposed S-Box

0.5000	0.5313	0.5469	0.5000	0.5156	0.5000	0.4844	0.4687
0.5156	0.5156	0.4844	0.5156	0.5000	0.4844	0.4688	0.5156
0.5313	0.5312	0.5781	0.5625	0.5312	0.5000	0.4219	0.4531
0.5000	0.4688	0.4063	0.5000	0.5469	0.5313	0.4531	0.4844
0.5000	0.5000	0.4688	0.4844	0.5000	0.5313	0.5000	0.5625
0.5156	0.4531	0.5313	0.5313	0.5469	0.5156	0.4844	0.5313
0.4844	0.4688	0.5000	0.5000	0.4844	0.4844	0.5000	0.5000
0.5156	0.5000	0.4844	0.5156	0.5313	0.5000	0.5625	0.5625

the ideal value 0.5 for a secure S-Box. In general, the SAC of an S-Box is represented by the dependence matrix P and each element P_{ij} of the dependence matrix is expected to close to the ideal value 0.5. The offset value of SAC can be calculated by the following formula:

$$\frac{1}{n^2} \sum_{1 \le i \le n} \sum_{1 \le j \le m} | \, 0.5 - P_{ij} \, | \, .$$

The dependent matrix of the proposed S-Box is given in Table 2. The SAC mean value is 0.5046 and the generated S-Box exhibited an offset value of 0.02417.

6.3 Differential approximation probability (DP)

The differential approximation probability is used to measure the XOR distribution balance between input and output bits of S-Box. A lower DP value provides a strong resistance to the differential attack. One can calculate the differential uniformity of a map f as follows [11]:

$$DP(f) = \max_{\Delta_x \neq 0, \Delta_y} \left(\frac{\#\{x \in GF(2^n) | f(x) \oplus f(x + \Delta_x) = \Delta_y\}}{2^n} \right),$$

here Δ_x and Δ_y are called input difference and output difference, respectively. The DP matrix of the proposed S-Box is presented in Table 3.

6.4 Bit independence criterion (BIC)

The bit independent criterion (BIC) is used to measure the level of dependent change in any pair of output bits when any input bit is retreated. A BIC-SAC value closer to 0.5 and a larger BIC-non-linearity value provides a better performance for an S-Box, [16]. The BIC-SAC and BIC nonlinearity values of our S-Box are presented in Tables 4 and 5, respectively. For the proposed S-Box, the minimum and maximum values of the BIC-SAC are 0.4629 and 0.5156, and the mean value is 0.5005. The minimum and maximum values of the BIC-non-linearity are 100 and 108, and the mean value is 104 These results are tabulated in Table 4. It can be seen from this table that the S-Box has a good performance in this criterion.

6.5 Linear approximation probability (LP)

Linear approximation probability (LP) is used to predict the imbalance of an event observed at output bits in response to changes at input bits [1, 33]. A lower LP value is desired to

Table 3 The DP matrix of the proposed *S*-Box

8	6	6	6	6	8	8	8	6	6	10	8	8	6	8	6
8	8	8	6	10	6	4	6	4	8	6	6	6	8	8	6
6	6	8	6	6	4	6	6	8	8	8	6	6	6	6	6
6	8	6	8	6	6	6	8	6	8	8	6	8	6	6	6
6	8	8	6	6	6	10	6	8	8	10	8	10	8	8	6
6	8	8	6	4	6	6	10	8	6	8	6	6	6	6	10
6	6	8	8	6	6	8	6	6	6	8	8	6	6	6	6
6	10	8	6	6	6	6	10	8	8	8	6	8	6	8	6
8	10	8	10	8	8	8	8	8	8	6	6	6	8	6	8
4	10	6	8	6	6	8	8	6	8	8	6	8	10	8	8
6	6	6	6	8	6	8	6	8	4	6	6	6	6	8	8
6	6	6	8	6	6	6	8	6	6	6	8	8	8	6	6
6	8	6	8	6	6	6	8	8	6	6	6	8	6	6	6
6	8	6	6	8	6	8	8	8	6	6	6	8	6	8	6
8	6	6	6	6	8	6	8	6	6	6	6	8	6	6	10
8	6	6	6	8	8	8	6	8	8	6	6	6	6	8	0

resist to linear attacks. The linear approximation probability is calculated by the following formula:

$$LP = \max_{\Gamma_x \Gamma_y \neq 0} \left| \frac{\#\{x/x \cdot \Gamma_x = S(x) \cdot \Gamma_y = \Delta_y\}}{2^n} - \frac{1}{2} \right|,$$

where x is the collection of all possible inputs, and 2^n is the total number of elements. The Γ_x and Γ_y are the required masks which operate on the parity of both input and output bits, respectively. The maximum value of the LP for our proposed *S*-Box is 0.16406.

6.6 Comparison of performance of proposed *S*-Box

In this section, we compare our *S*-Box with other *S*-Boxes generated by some of the latest algorithms to show the superiority of our *S*-Box. Table 6 presents the comparison results in terms of non-linearity, differential and linear approximation probabilities, strict avalanche and bit independence criteria with [2, 4–6, 12, 22, 25, 27, 42, 46, 53, 54].

According to Table 6, the comments are as follows:

Table 4 BIC-SAC matrix of the proposed *S*-Box

0	0.5039	0.5020	0.4961	0.5156	0.4922	0.5059	0.5078
0.5039	0	0.4629	0.5098	0.5137	0.5156	0.4961	0.5059
0.5020	0.4629	0	0.5117	0.5156	0.5000	0.4981	0.4883
0.4961	0.5098	0.5117	0	0.5000	0.4805	0.4863	0.4902
0.5156	0.5137	0.5156	0.5000	0	0.5098	0.4883	0.5000
0.4922	0.5156	0.5000	0.4805	0.5098	0	0.5098	0.5059
0.5059	0.4961	0.4981	0.4863	0.4883	0.5098	0	0.5020
0.5078	0.5059	0.4883	0.4902	0.5000	0.5059	0.5020	0

Table 5 BIC-non-linearity matrix of the proposed *S*-Box

0	106	106	106	104	100	106	104
106	0	102	104	100	104	108	104
106	102	0	104	106	106	102	104
106	104	104	0	104	102	100	108
104	100	106	104	0	106	100	104
100	104	106	102	106	0	102	104
106	108	102	100	100	102	0	106
104	104	104	108	104	104	106	0

- It is known that *S*-Box with higher nonlinearity value is more secure than that with smaller nonlinearity value and it can be seen from the Table 6 that the proposed S-box has higher nonlinearity values compared to the others as aimed.
- The optimum value is 0.5 for the strict avalanche criterion. It is seen from the Table 6 that the SAC value of the proposed *S*-Box is closer to optimum value 0.5 comparing to [4–6, 22, 27, 42, 46, 54].
- It is also desired that the nonlinearity value should be as large as possible and the strict avalanche criterion is close to 0.5 with respect to the output bits independence criterion. It can be seen from the Table 6 that proposed *S*-Box has better BIC nonlinearity and BIC SAC value comparing to [5, 12, 42, 46]. In particular, BIC SAC value of the proposed *S*-Box has closer optimum value than the [2, 5, 6, 12, 22, 27, 42, 46, 54].

7 An application of generated *S*-Box to an RGB color image encryption

In this section, we briefly explain how we apply the proposed *S*-Box to image encryption, the details of the encryption scheme can be found in [20]. An image encryption algorithm

Table 6 The comparison results of *S*-Boxes generated by different methods

S-Box	Nonlinearity			Bit Independence Criteria (BIC)		SAC	DP
	min	max	avg	Nonlinearity	SAC		
Proposed *S*-Box	108	112	**110**	104	0.5005	0.5046	10
Ahmed et al. [4]	106	108	107.5	104.3	0.5001	0.4944	10
Ahmad et al. [2]	106	110	107	105.5	0.5010	0.5015	10
Alhadawi et al. [6]	106	108	107	104.6	0.4974	0.496	10
Alhadawi et al. [5]	106	110	108.5	103.85	0.5011	0.4995	10
Chen [12]	102	106	104	103.2	0.4971	0.4980	10
Farah et al. [22]	104	110	106.5	105.2	0.4984	0.5120	10
Hematpour and Ahadpour [25]	104	108	106.5	105.85	0.4995	0.5036	10
Hussain et al. [27]	100	108	104.7	105	0.4965	0.4037	32
Tian and Lu [42]	106	110	107.5	103.7	0.5025	0.5093	10
Wang et al. [46]	108	108	108	90	0.4950	0.5068	10
Zamli et al. [53]	106	110	109.25	104.07	0.5005	0.5017	10
Zamli [54]	106	112	109.75	104.35	0.5009	0.5068	10

consists of four stages: generation of S-Box from shuffling algorithm, generation of the main key by using secret and public keys, S-Box based permutation and diffusion phases. S-Box based permutation and diffusion phases are the same as in [20]. In the permutation phase we use the chaotic map from [43]. The secret key, which is used in the main key generation step, differs from [20]. The secret key in our proposed encryption algorithm depends on a time signature (h, m, s) and the proposed S-Box. More precisely, we first take the time signature then compute the sum of the hour and minute values in modulo 16 and compute the second value in modulo 2. The sum of the hour and minute values in modulo 16 determines which row or column of the S-Box will be taken in the secret key. If the second value in modulo 2 is 0 then we choose row of the S-Box otherwise we choose the column of the S-Box. A secret key $s_k = (h, m, s, b_0, b_1, \cdots, b_{15})$, where b_i denotes the i-th row or column of the S-Box, is formed by combining SHA-512 and MD5 hash values of s_k. Then, a public key is produced from the plaintext image using the combination of SHA-512 and MD5 hash values. Finally, the main key is acquired via xor operation between the public and secret key. After the generation of main key, the S-Box-based diffusion operation is applied to the plain text image in order to change the pixel values of the plain text. Then, in the permutation phase, we use values of the proposed S-Box and also the main key to obtain initial values and control parameter for the chaotic map.

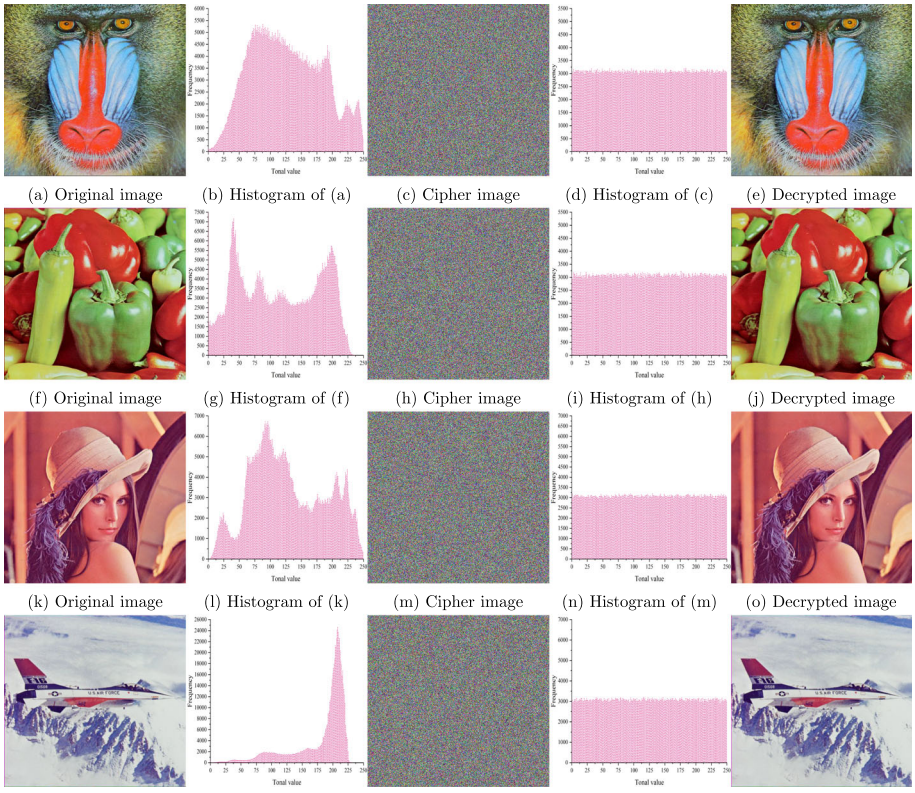
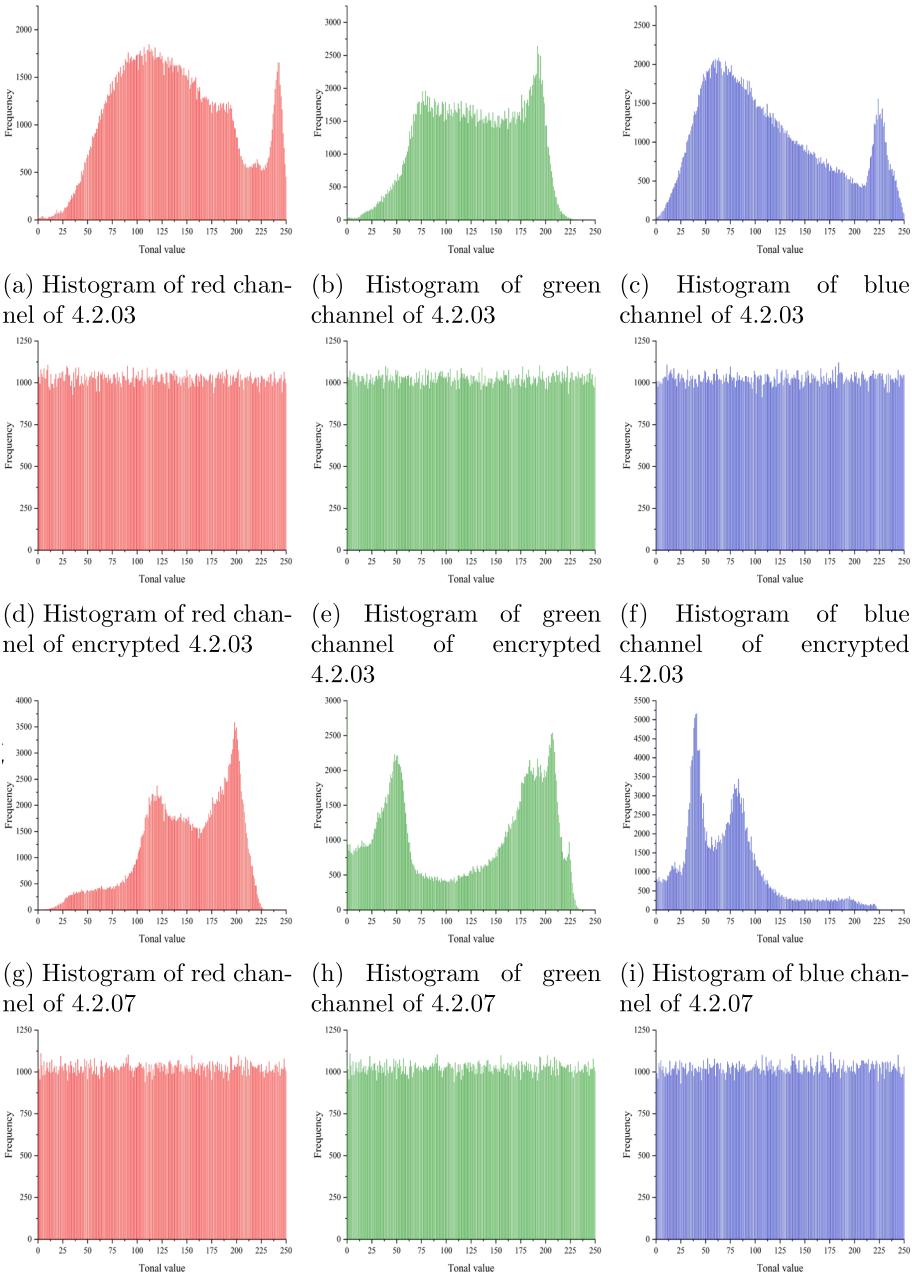


Fig. 3 Simulation results of the proposed image encryption algorithms



(a) Histogram of red channel of 4.2.03

(b) Histogram of green channel of 4.2.03

(c) Histogram of blue channel of 4.2.03

(d) Histogram of red channel of encrypted 4.2.03

(e) Histogram of green channel of encrypted 4.2.03

(f) Histogram of blue channel of encrypted 4.2.03

(g) Histogram of red channel of 4.2.07

(h) Histogram of green channel of 4.2.07

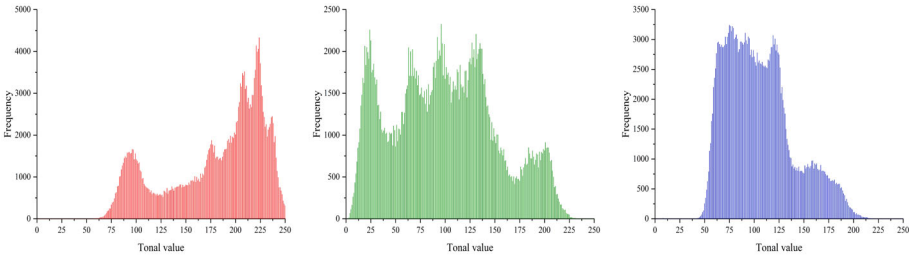
(i) Histogram of blue channel of 4.2.07

(j) Histogram of red channel of encrypted 4.2.07

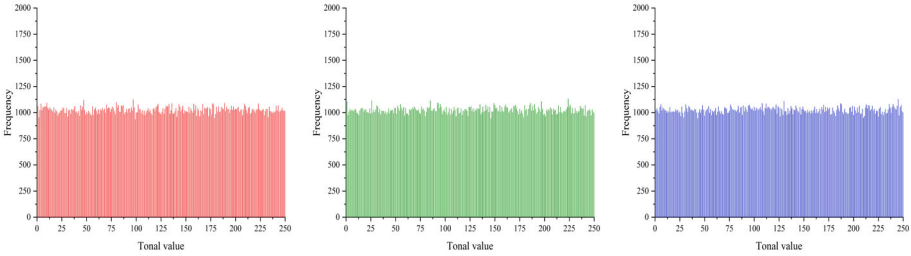
(k) Histogram of green channel of encrypted 4.2.07

(l) Histogram of blue channel of encrypted 4.2.07

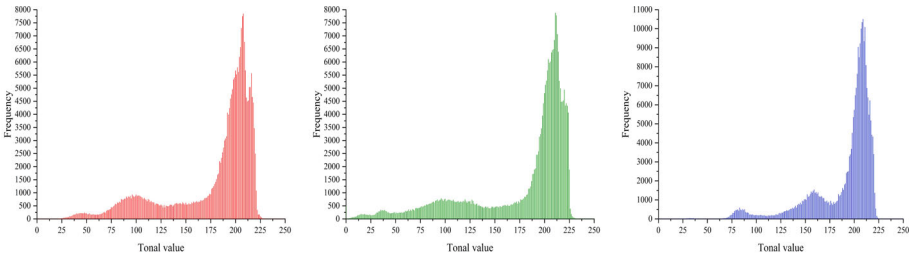
Fig. 4 Histograms of plaintext and cipher text images of 4.2.03 and 4.2.07



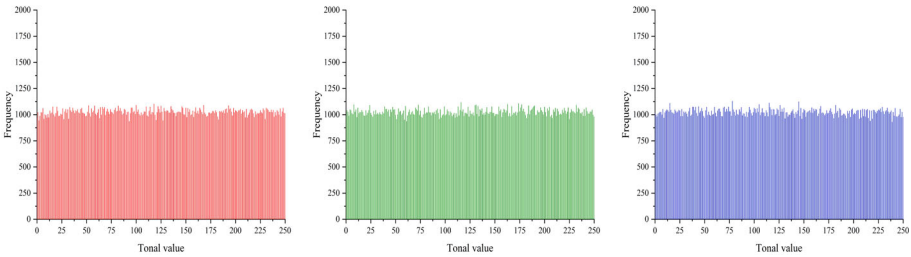
(a) Histogram of red channel of 4.2.04 (b) Histogram of green channel of 4.2.04 (c) Histogram of blue channel of 4.2.04



(d) Histogram of red channel of encrypted 4.2.04 (e) Histogram of green channel of encrypted 4.2.04 (f) Histogram of blue channel of encrypted 4.2.04



(g) Histogram of red channel of 4.2.05 (h) Histogram of green channel of 4.2.05 (i) Histogram of blue channel of 4.2.05



(j) Histogram of red channel of encrypted 4.2.05 (k) Histogram of green channel of encrypted 4.2.05 (l) Histogram of blue channel of encrypted 4.2.05

Fig. 5 Histograms of plaintext and cipher text images of 4.2.04 and 4.2.05

8 Experimental results and security analysis

Image encryption experiments are performed by using 512 × 512 images of 4.2.07, 4.2.03, 4.2.04 and 4.2.05 taken from the website (https://sipi.usc.edu/database/). The encryption results were evaluated on a workstation with Intel Xeon 4.0 GHz processor and 64 GByte RAM using MATLAB software.

8.1 Simulation results

In this section, simulation results are presented for images of 4.2.03.jpg (512 × 512), 4.2.07.jpg (512 × 512), 4.2.04.jpg (512 × 512) and 4.2.05.jpg (512 × 512) in Fig. 3. It can be seen from Fig. 3 that the cipher images don't contain any meaningful information and they are recovered successfully by decryption processes (Figs. 4 and 5).

8.1.1 Ability of resisting differential attack

There are two criteria to test the ability of an encryption algorithm in resisting the differential attack; the number of pixel change rate (NPCR) and unified average changing intensity (UACI) [44]. When the two images are completely different, the expected value of NPCR is 99.6094% and expected value of UACI is 33.4635% [49]. The NPCR and UACI results are given in Table 7.

8.2 Information entropy analysis

The entropies of the ciphered images are measured to evaluate their uncertainties [39]. The entropy value, which is close to 8, is desired for an efficient encryption algorithm. The information entropy of sequence s can be calculated as follows:

$$H(s) = \sum_{i=1}^{l-1} P(s_i) \log_2 \frac{1}{P(s_i)},$$

here $l = 2^m$, and m is the total number of samples, s_i is the gray value of the image, $P(s_i)$ denotes the probability of the element s_i in the sequence s, and $\sum_{i=1}^{l-1} P(s_i) = 1$. The results of entropy analysis are presented in Table 8.

8.2.1 Noise attack

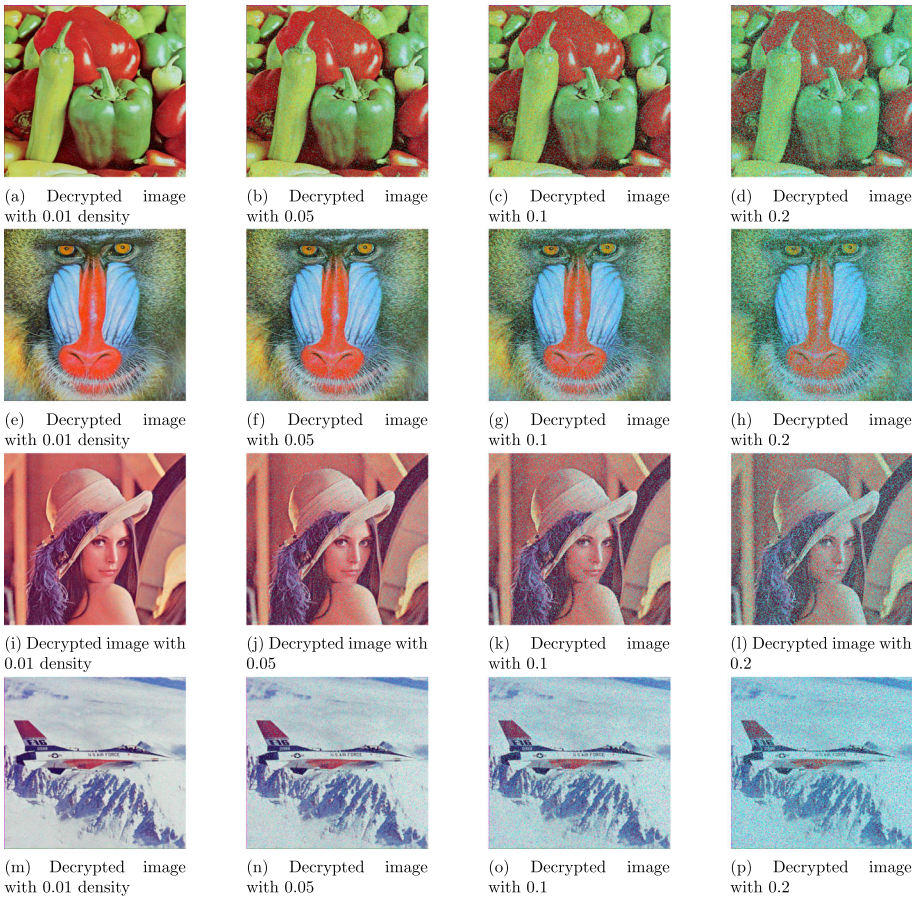
A noise attack is used to prevent the receiver from decrypting cipher text information successfully by ruining the integrity of the cipher text information. Salt and pepper noise (SPN) is inserted to the cipher text image to analyze this attack. In our experiment, different densities noise attacks are simulated. In the experiment, SPN densities

Table 7 NPCR and UACI analysis

Image/H(s)	4.2.03	4.2.07	4.2.04	4.2.05
NPCR	99.0115%	98.8123%	99.4099%	99.2110%
UACI	33.2698%	33.1807%	33.3857%	33.3196%

Table 8 Comparison of information entropy analysis

Image/H(s)	Channel	Plain image	Encrypted image	[18]	[41]	[55]	[10]	[51]
4.2.03	R	7.70677	7.99927	7.999296	-	7.99141	7.9973	-
(512 × 512)	G	7.47443	7.99933	7.999292	-	7.99146	7.9974	-
	B	7.75222	7.99930	7.999258	-	7.99153	7.9975	-
Average		7.64444	7.99930	7.9993	-	7.9915	7.9974	-
4.2.07	R	7.33883	7.99933	7.999382	7.9915	7.99114	7.9972	7.99930
(512 × 512)	G	7.49625	7.99928	7.999348	7.9914	7.99123	7.9969	7.99929
	B	7.05831	7.99927	7.999251	7.9912	7.99150	7.9973	7.99920
Average		7.29780	7.99929	7.9993	7.9914	7.9912	7.9971	7.9993
4.2.04	R	7.25309	7.99931	7.999290	7.9912	7.99171	7.9951	7.99932
(512 × 512)	G	7.59515	7.99933	7.999477	7.9913	7.99121	7.9965	7.99931
	B	7.96852	7.99936	7.999319	7.9914	7.99177	7.9829	7.99935
Average		7.27225	7.99933	7.9994	7.9913	7.9916	7.9915	7.9993
4.2.05	R	6.71777	7.99934	7.999310	-	-	7.992	-
(512 × 512)	G	6.79898	7.99927	7.999309	-	-	7.9957	-
	B	6.21377	7.99927	7.999330	-	-	7.9968	-
Average		6.57684	7.99929	7.9993	-	-	7.9948	-



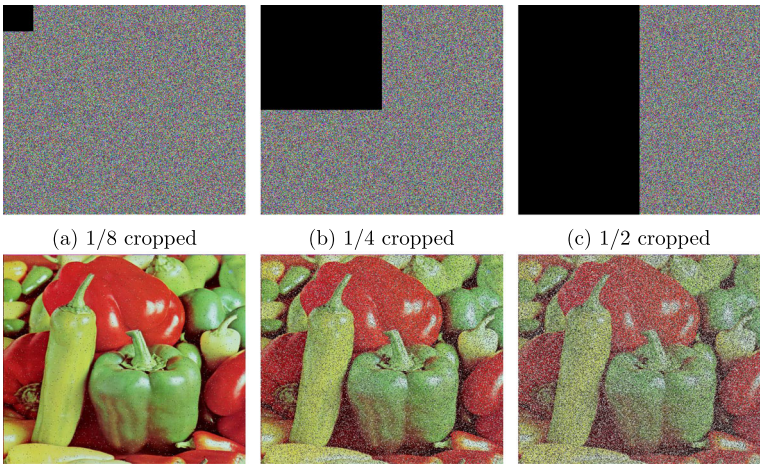
(a) Decrypted image with 0.01 density
(b) Decrypted image with 0.05
(c) Decrypted image with 0.1
(d) Decrypted image with 0.2
(e) Decrypted image with 0.01 density
(f) Decrypted image with 0.05
(g) Decrypted image with 0.1
(h) Decrypted image with 0.2
(i) Decrypted image with 0.01 density
(j) Decrypted image with 0.05
(k) Decrypted image with 0.1
(l) Decrypted image with 0.2
(m) Decrypted image with 0.01 density
(n) Decrypted image with 0.05
(o) Decrypted image with 0.1
(p) Decrypted image with 0.2

Fig. 6 Cipher text images with inserting different SPN densities

of 0.01, 0.05, 0.1, 0.2 are inserted to the cipher text images of 4.2.07, 4.2.03, 4.2.04 and 4.2.05. The decrypted results of the cipher text can be seen from Fig. 6. The PSNR values of 4.2.07 are measured as 22.6490, 15.7299, 12.7526, 9.8134 for the decipher images with SPN densities of 0.01, 0.05, 0.1, 0.2, respectively. The PSNR values of 4.2.03 are measured as 27.7881, 16.9301, 13.9620, 11.0236 for the decipher images with SPN densities of 0.01, 0.05, 0.1, 0.2, respectively. The PSNR values of 4.2.04 are measured as 22.7631, 15.7681, 12.7978, 9.8819 for the decipher images with SPN densities of 0.01, 0.05, 0.1, 0.2, respectively. The PSNR values of 4.2.05 are measured as 24.6235, 17.6617, 14.7365, 11.8484 for the decipher images with SPN densities of 0.01, 0.05, 0.1, 0.2, respectively. We understand from Fig. 6 that the proposed encryption and decryption algorithm resists the noise attacks effectively.

8.2.2 Cropping attack

A cropping attack is a tool that is used by attackers in order to disturb the integrity of the cipher text information. The goal of this attack is to prevent the receiver to decrypt the cipher



(a) 1/8 cropped (b) 1/4 cropped (c) 1/2 cropped

(d) Decrypted image of (a) (e) Decrypted image of (b) (f) Decrypted image of (c)

Fig. 7 Cropping attack analysis of the proposed encryption for the 4.2.07

text successfully. Minimum corruption is expected from a strong and robust image encryption algorithm. Different degrees of cropping attack are simulated in Fig. 7. In the experiment, the decryption results of the cropped decrypted 4.2.07.jpg (512 × 512) for ratios 1/2, 1/4, and 1/8 are given in Fig. 7. As a consequence, identifiable decrypted images are obtained after the decryption process.

Moreover, the PSNR scores of the decipher images are listed in Table 9. Higher PSNR values corresponds to lower corruption. The PSNR value is calculated as follows [26, 47]:

$$PSNR := 10 \log \left(\frac{255^2}{MSE} \right),$$

where MSE stands for the mean-squared error and it is given by the following formula:

$$MSE := \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (e_{ij} - f_{ij})^2.$$

Here $E := (e_{ij})$ is the plaintext image and $F := (f_{ij})$ is the decrypted image with cropping.

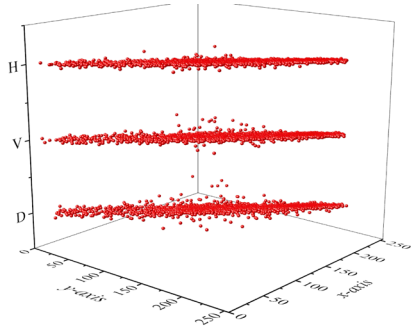
Table 9 The PSNR scores for the decrypted images under-cropping attack

Cropping ratio	4.2.03	4.2.07	4.2.04	4.2.05
1/8	26.7951	26.2603	26.7467	25.8812
1/4	14.7892	14.0901	14.6463	13.9892
1/2	11.7878	11.0810	11.6399	10.9852

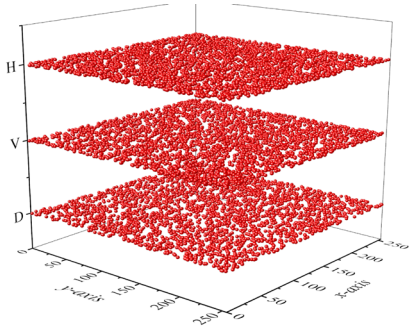
8.2.3 Correlation analysis of adjacent pixels

The correlation coefficient between two adjacent pixels in an image is calculated by the following formula:

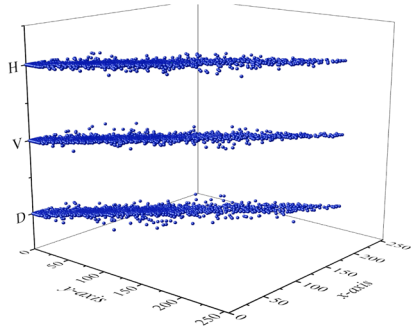
$$r_{\alpha\beta} = \frac{cov(\alpha, \beta)}{\sqrt{\psi(\alpha)}\sqrt{\psi(\beta)}},$$



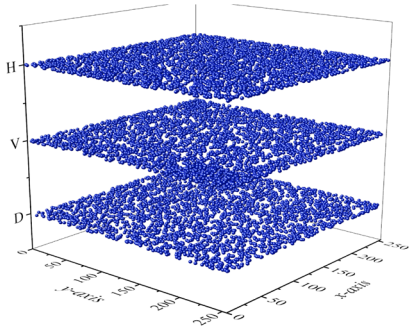
(a) Pixel distribution of plaintext image for red channel



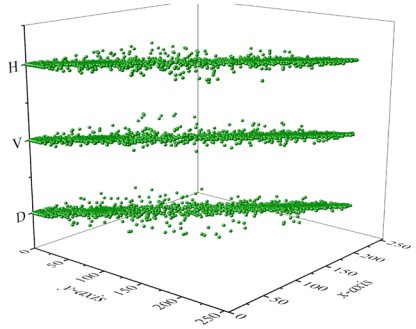
(b) Pixel distribution of cipher text image for red channel



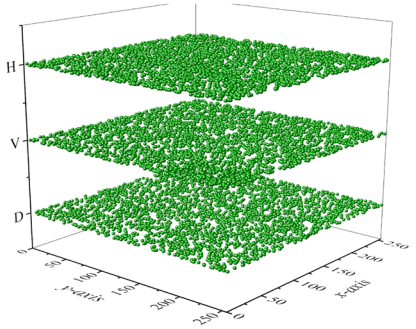
(c) Pixel distribution of plaintext image for blue channel



(d) Pixel distribution of cipher text image for blue channel



(e) Pixel distribution of plaintext image for green channel



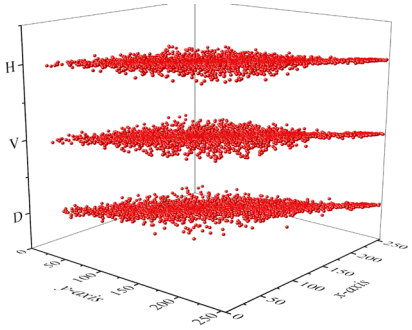
(f) Pixel distribution of cipher text image for green channel

Fig. 8 Pixel distribution of plaintext and cipher text image of 4.2.07 in horizontal, vertical and diagonal directions (H, V, D denotes the horizontal, vertical and diagonal directions respectively)

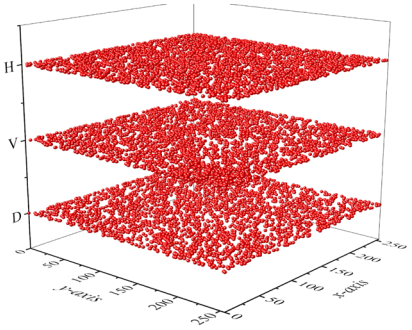
where

$$\psi(\alpha) = \frac{1}{N}\sum_{i=1}^{N}[\alpha_i - E(\alpha)]^2,$$

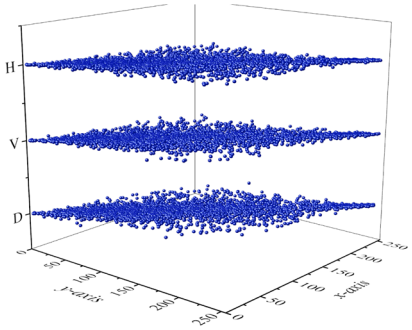
$$cov(\alpha, \beta) = \frac{1}{N}\sum_{i=1}^{N}[\alpha_i - E(\alpha)][\beta_i - E(\beta)].$$



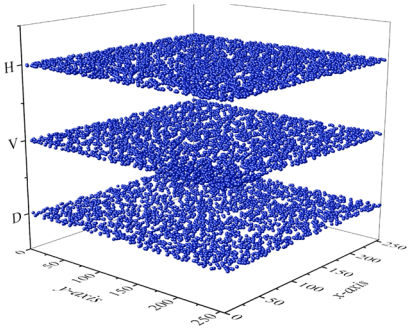
(a) Pixel distribution of plaintext image for red channel



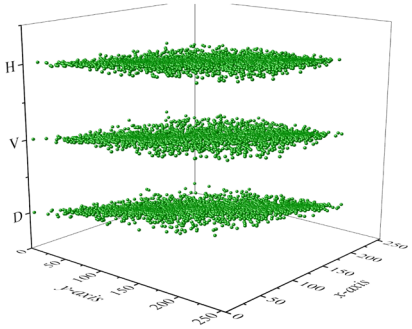
(b) Pixel distribution of cipher text image for red channel



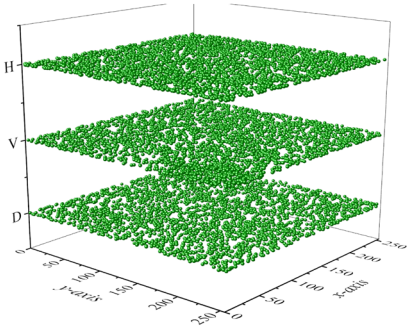
(c) Pixel distribution of plaintext image for blue channel



(d) Pixel distribution of cipher text image for blue channel



(e) Pixel distribution of plaintext image for green channel



(f) Pixel distribution of cipher text image for green channel

Fig. 9 Pixel distribution of plaintext and cipher text image of 4.2.03 in horizontal, vertical and diagonal directions (H, V, D denotes the horizontal, vertical and diagonal directions respectively)

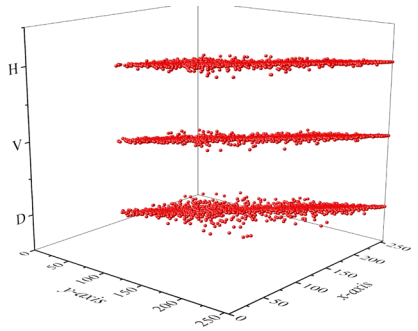
Here, α_i and β_i denote two adjacent pixels, N is the total number of duplets (α_i, β_i) obtained from the image; $E(\alpha)$ and $E(\beta)$ are the mean values of α_i and β_i, respectively. The correlation between adjacent pixels are calculated as following; first $N = 1000$ pairs of adjacent pixels are selected from the image, and then their correlation coefficient is calculated. If the value of $r_{\alpha\beta}$ is close to 0, then the encryption algorithm works well.

The relationship between two variables or pixels in an image is measured by correlation analysis. There exists a high correlation in adjacent pixels of the image in the horizontal, vertical, and diagonal directions for a plaintext. Decreasing the correlation of two adjacent pixels increases the resistance of an image encryption algorithm (Figs. 8 and 9).

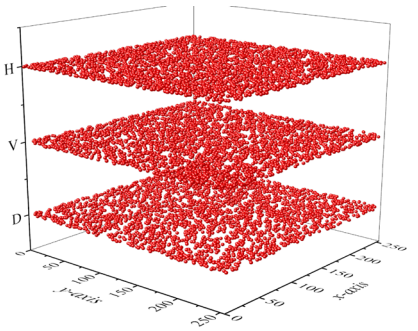
The correlations of two adjacent pixels in the plain images and their encrypted images are given in Table 10. Moreover, the pixel distributions of adjacent pixels in the three directions of the original and ciphered images are given in Figs. 10 and 11.

Table 10 Comparison of correlation coefficients

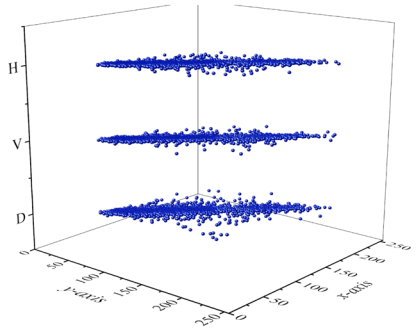
	Channel	Direction	4.2.03 (512 × 512)	4.2.07 (512 × 512)	4.2.04 (512 × 512)	4.2.05 (512 × 512)
Our scheme	R	Horizontal	−0.000697	−0.000335	−0.000550	−0.000496
	R	Vertical	0.000559	0.000142	−0.000353	0.000652
	R	Diagonal	−0.000287	0.000741	0.000367	0.000883
	G	Horizontal	0.000954	−0.000411	−0.000418	0.000627
	G	Vertical	0.000416	0.000425	−0.000189	0.000884
	G	Diagonal	−0.000456	0.000357	0.000687	−0.000214
	B	Horizontal	−0.000183	0.000736	−0.000298	−0.000517
	B	Vertical	0.000171	−0.000892	−0.000097	0.000898
	B	Diagonal	−0.000999	0.000138	−0.000396	−0.000359
Demirtas [18]	R	Horizontal	−0.000022	0.000960	−0.0040	−0.0021
	R	Vertical	−0.000820	−0.000880	0.0015	−0.0032
	R	Diagonal	−0.000510	−0.000300	0.0025	0.0040
	G	Horizontal	−0.000022	0.000960	0.0074	−0.0024
	G	Vertical	−0.000820	−0.000880	−0.0016	0.0022
	G	Diagonal	−0.000510	−0.000300	−0.0024	0.0083
	B	Horizontal	−0.000022	0.000960	−0.0002	−0.0047
	B	Vertical	−0.000820	−0.000880	−0.0041	−0.0009
	B	Diagonal	−0.000510	−0.000300	0.0011	−0.0035
Li et al. [37]	R	Horizontal	−0.000022	0.000960	−0.0022	0.0006
	R	Vertical	−0.000820	−0.000880	0.0009	0.0013
	R	Diagonal	−0.000510	−0.000300	0.0013	0.0016
	G	Horizontal	−0.000022	0.000960	0.0057	0.0023
	G	Vertical	−0.000820	−0.000880	−0.0041	0.0010
	G	Diagonal	−0.000510	−0.000300	0.0017	−0.0057
	B	Horizontal	−0.000022	0.000960	0.00007	0.0009
	B	Vertical	−0.000820	−0.000880	0.00004	−0.0017
	B	Diagonal	−0.000510	−0.000300	0.0104	0.0083



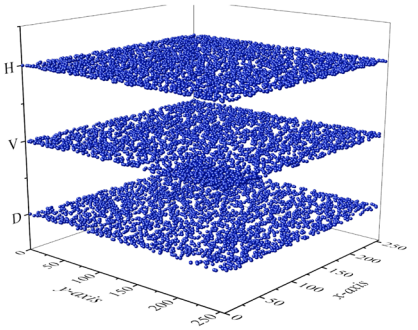
(a) Pixel distribution of plaintext image for red channel



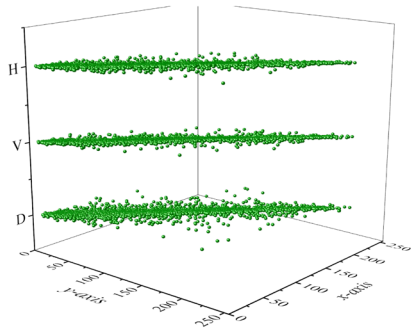
(b) Pixel distribution of cipher text image for red channel



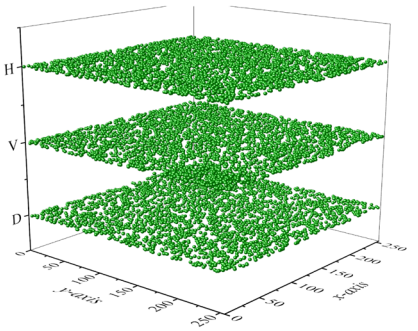
(c) Pixel distribution of plaintext image for blue channel



(d) Pixel distribution of cipher text image for blue channel



(e) Pixel distribution of plaintext image for green channel

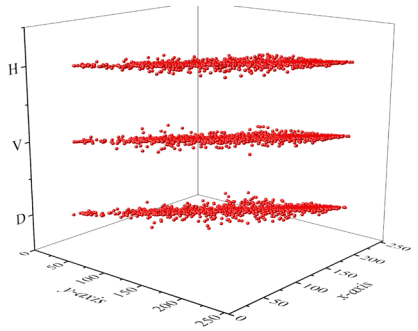


(f) Pixel distribution of cipher text image for green channel

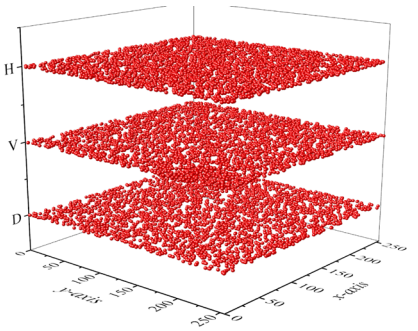
Fig. 10 Pixel distribution of plaintext and cipher text image of 4.2.04 in horizontal, vertical and diagonal directions (H, V, D denotes the horizontal, vertical and diagonal directions respectively)

8.2.4 Key space

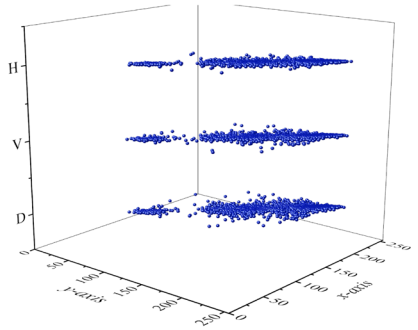
A brute force attack is a hacking method that predicts the key by trying numerous passwords. An image algorithm scheme with a short key is not strong enough at resisting exhaustive brute-force attacks. An image algorithm with the key space larger than 2^{100} is strong enough to



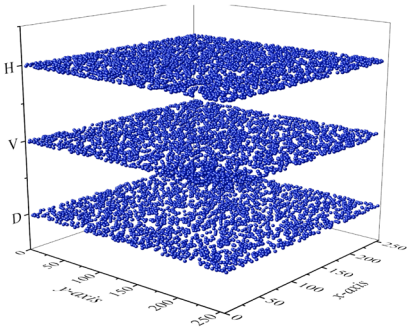
(a) Pixel distribution of plaintext image for red channel



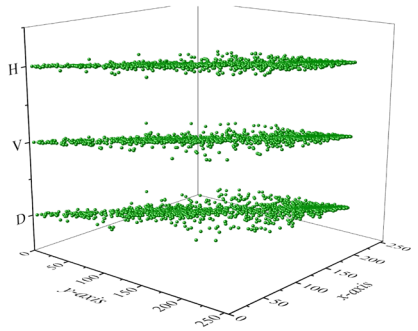
(b) Pixel distribution of cipher text image for red channel



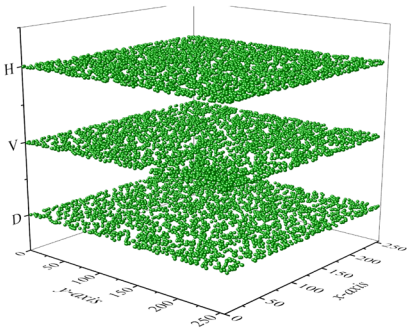
(c) Pixel distribution of plaintext image for blue channel



(d) Pixel distribution of cipher text image for blue channel



(e) Pixel distribution of plaintext image for green channel



(f) Pixel distribution of cipher text image for green channel

Fig. 11 Pixel distribution of plaintext and cipher text image of 4.2.05 in horizontal, vertical and diagonal directions (H, V, D denotes the horizontal, vertical and diagonal directions respectively)

resist brute-force attacks [31]. The secret key in our proposed encryption algorithm depends on a time signature (h, m, s) and the row or column of the S-Box. A secret key $s_k = (h, m, s, b_0, b_1, \cdots, b_{15})$, where b_i denotes the i- th row or column of the S-Box, is formed by combining SHA-512 and MD5 hash values of s_k.Therefore the key space of the secret

key in our proposed encryption algorithm can be calculated as follows:

$$24(60)(60)(256)(255)\cdots(241) = 1.82(10)^{43} \approx 2^{143},$$

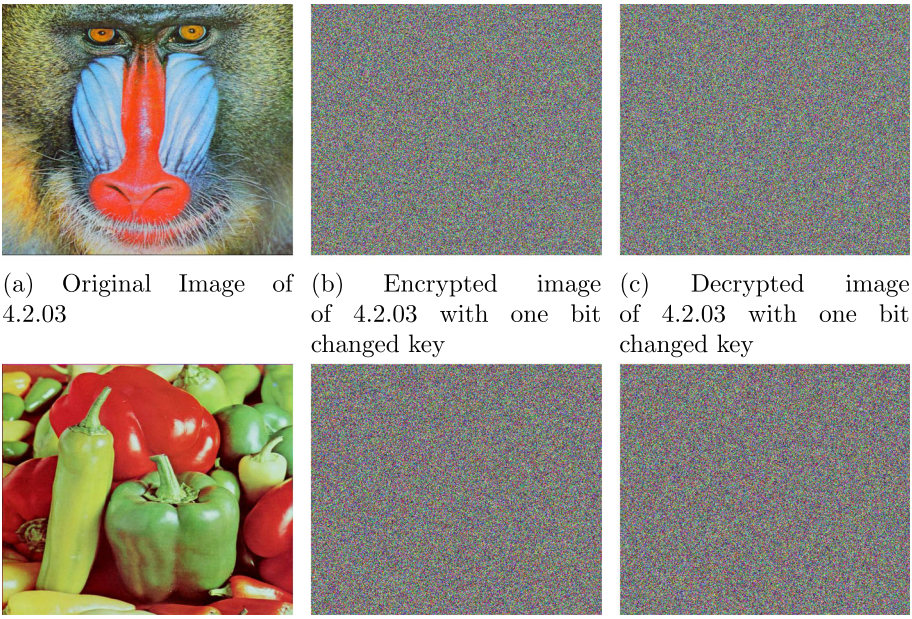
which is greater than 2^{100}, and obviously it can resist an exhaustive attack.

8.2.5 Key sensitivity

An effective encryption scheme should be sensitive to any bit flipping in the secret key to provide security against brute-force attacks. We examine the key sensitivity by changing one bit of the original key. In addition to that, NPCR and UACI values are calculated to test for the key sensitivity of the encryption and decryption processes.

It is obvious from Fig. 12 that decrypted images of Mandrill and Peppers with one bit changed keys are very complicated and it is not possible to understand what the plaintext images are from these decrypted images. As it is explained in the above, NPCR and UACI values are also used to test the sensitivity of the encryption and decryption processes. NPCR and UACI values of the encrypted (decrypted) images are obtained by using one bit changed keys and these values are compared with the original encrypted (decrypted) images. The results are presented in Table 11.

These results straightforwardly indicate that the proposed encryption scheme is very sensitive to its secret key.



(a) Original Image of 4.2.03 (b) Encrypted image of 4.2.03 with one bit changed key (c) Decrypted image of 4.2.03 with one bit changed key

(d) Original Image of 4.2.07 (e) Encrypted image of 4.2.07 with one bit changed key (f) Decrypted image of 4.2.07 with one bit changed key

Fig. 12 Decrypted and encrypted images of 4.2.03 and 4.2.07 with one bit changed keys

Table 11 NPCR and UACI analysis for key sensitivity

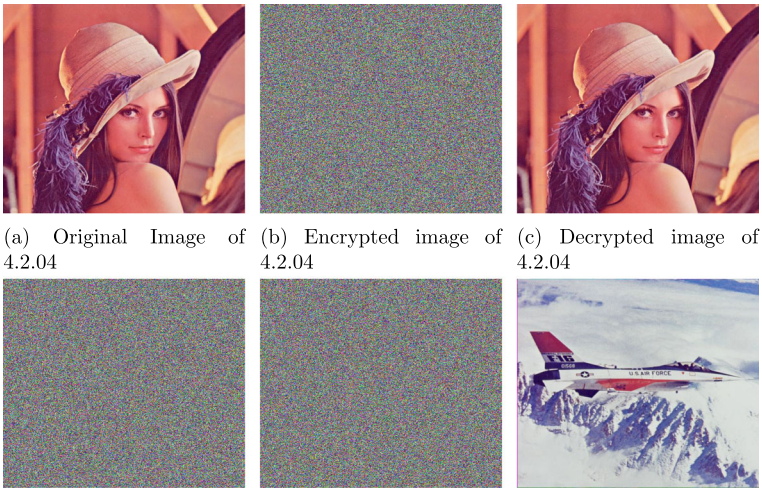
Image/H(s)	4.2.03	4.2.07
NPCR for Encryption	99.6019%	99.6165%
UACI for Encryption	33.0345%	33.6441%
NPCR for Decryption	99.6164%	99.6196%
UACI for Decryption	33.1128%	33.6689%

8.2.6 Known plaintext attacks

The known plaintext attack (KPA) is an attack model in which the attacker has a sufficient number of plain/ ciphertext pairs encrypted by a fixed secret key. These pairs are used to decrypt the other ciphers which are encrypted by the same secret key. NPCR and UACI values of decrypted 4.2.05 with the secret key obtained by KPA are 99.6172 and 32.6293, respectively (Fig. 13).

9 Conclusion

In this study, the pseudo-random sequences which is used in cryptographic applications is constructed by exponentiation in finite fields and these sequences are used for enhancing the security and efficacy of the S-Box in our encryption scheme. A shuffling algorithm that based on the pseudo-random sequences is also developed to enhance the performance of the S-Box. *S*-Box test results demonstrate that proposed *S*-Box expose better performance than that of previous studies and it is possible to make stronger *S*-box designs by using pseudo-random sequences. In addition to that, the RGB color images are encrypted by using proposed S-Box



(a) Original Image of 4.2.04 (b) Encrypted image of 4.2.04 (c) Decrypted image of 4.2.04

(d) Encrypted 4.2.05 with the correct secret key (e) Decrypted 4.2.05 with the secret key obtained by KPA (f) Decrypted 4.2.05 with the correct secret key

Fig. 13 Decrypted images by the known-plaintext attack

to see the effectiveness of it in applications. The simulation results, security analyses and comparative analyses confirm the reliability and robustness of the proposed S-Box for image encryption.

Data Availability This work does not have any dataset.

Declarations

Conflict of Interest The authors declare that they have no conflict of interest.

References

1. Aboytes-Gonzalez JA, Murguia JS, Mejia-Carlos Gonzalez-Aguilar MH, Ramirez-Torres MT (2018) Design of a strong *S*-box based on a matrix approach. Nonlinear Dyn 94:(3)2003-2012
2. Ahmad M, Bhatia D, Hassan Y (2015) A novel ant colony optimization based scheme for substitution box design. Procedia Comput Sci 57:572–580
3. Ahmad M, Alkanhel R, El-Shafai W, Algarni AD, El-Samie FEA, Soliman NF (2022) Multi-Objective Evolution of Strong S-Boxes Using Non-Dominated Sorting Genetic Algorithm-II and Chaos for Secure Telemedicine IEEE. Access 10:112757–112775. https://doi.org/10.1109/ACCESS.2022.3209202
4. Ahmed HA, Zolkipli MF, Ahmad M (2019) A novel efficient substitution-box design based on firefly algorithm and discrete chaotic map. Neural Comput Appl 31(11):7201–7210
5. Alhadawi HS, Majid MA, Lambic D, Ahmad M (2021) A novel method of *S*-box design based on discrete chaotic maps and cuckoo search algorithm. Multimed Tools Appl 80(5):7333–7350
6. Alhadawi HS, Lambic D, Zolkipli MF, Ahmad M (2020) Globalized firefly algorithm and chaos for designing substitution box. J Inf Secur Appl 55:102671
7. Ali TS, Ali R (2022) A novel color image encryption scheme based on a new dynamic compound chaotic map and S-box. Multimed Tools Appl 81:20585–20609. https://doi.org/10.1007/s11042-022-12268-6
8. Alsaif H, Guesmi R, Kalghoum A, Alshammari BM, Guesmi TA (2023) Novel Strong S-Box Design Using Quantum Crossover and Chaotic Boolean Functions for Symmetric Cryptosystems Symmetry 15:833. https://doi.org/10.3390/sym15040833
9. Artuger F, Ozkaynak F (2022) SBOX-CGA: substitution box generator based on chaos and genetic algorithm. Neural Comput Applic 34:20203–20211. https://doi.org/10.1007/s00521-022-07589-4
10. Basha SM, Mathivanan P, Ganesh AB (2022) Bit level color image encryption using Logistic-Sine-Tent-Chebyshev (LSTC) map. Optik 259:168956. https://doi.org/10.1016/j.ijleo.2022.168956
11. Biham E, Shamir A (1991) Differential Cryptanalysis of DES-like Cryptosystems In Menezes, A.J., Vanstone, S.A. (eds) Advances in Cryptology-CRYPTO' 90. CRYPTO 1990.Lecture Notes in Computer Science 537 Springer Berlin Heidelberg https://doi.org/10.1007/3-540-38424-31
12. Chen G (2008) A novel heuristic method for obtaining *S*-boxes. Chaos Solitons Fractals 36(4):1028–1036
13. Damg I (1998) On the randomness of Legendre and Jacobi sequences Advances in Cryptology Santa Barbara CA 163–172 Lecture Notes in Comput Sci 403 Springer. Berlin
14. Davenport H (1931) On the distribution of quadratic residues (mod p). J Lond Math Soc 6:49–54
15. Davenport H (1933) On the distribution of quadratic residues (mod p) II. J Lond Math Soc 6:46–52
16. Detombe J, Tavares S (1992) Constructing large cryptographically strong S-boxes: Advances in Cryptology Proc. of CRYPTO92 Lecture Notes in Computer Science 165-181
17. Deb S, Biswas B, Bhuyan B (2019) Secure image encryption scheme using high efficiency word-oriented feedback shift register over finite field. Multimed Tools Appl 78:34901–34925. https://doi.org/10.1007/s11042-019-08086-y
18. Demirtas M (2022) A new RGB color image encryption scheme based on cross-channel pixel and bit scrambling using chaos. Optik 265:169430. https://doi.org/10.1016/j.ijleo.2022.169430
19. Din M, Pal SK, Muttoo SK (2022) A new S-box design by applying Swarm Intelligence based technique. Int J Syst Assur Eng Manag 13:2963–2970. https://doi.org/10.1007/s13198-022-01766-3
20. Dougherty ST, Sahinkaya S, Ustun D (2023) A novel method for image encryption using time signature-dependent s-boxes based on latin squares and the playfair system of cryptography. Multimed Tools Appl. https://doi.org/10.1007/s11042-023-15240-0
21. El-Latif AAA, Abd-El-Atty B, Belazi A, Iliyasu AM (2021) Efficient chaos-based substitution-box and its application to image encryption. Electronics 10:(12)1392

22. Farah T, Rhouma R, Belghith S (2017) A novel method for designing *S*-box based on chaotic map and teaching- learning based optimization. Nonlinear Dyn 88(2):1059–1074
23. Gerardo de la Fraga L, Ovilla-Martínez B, (2023) A chaotic PRNG tested with the heuristic Differential Evolution. Integration 90:22–26. https://doi.org/10.1016/j.vlsi.2023.01.001
24. Haque AT, Abdulhussein A, Ahmad M, Waheed Falah M, Abd El-Latif AA (2022) A Strong Hybrid S-Box Scheme Based on Chaos, 2D Cellular Automata and Algebraic Structure. IEEE Access 10:116167–116181. https://doi.org/10.1109/ACCESS.2022.3218062
25. Hematpour N, Ahadpour S (2021) Execution examination of chaotic *S*-box dependent on improved PSO algorithm. Neural Comput Appl 33(10):5111–5133
26. Huynh-Thu Q, Ghanbari M (2008) Scope of validity of PSNR in image/video quality assessment. Electron Lett 44:(13)800–801 https://doi.org/10.1049/el:20080522
27. Hussain I, Shah T, Gondal MA, Khan WA, Mahmood H (2013) A group theoretic approach to construct cryptographically strong substitution boxes. Neural Comput Appl 23(1):97–104
28. Lai Q, Hu G, Erkan U, Toktas A (2023) High-efficiency medical image encryption method based on 2D Logistic-Gaussian hyperchaotic map. Appl Math Comput 442, 2023, https://doi.org/10.1016/j.amc.2022.127738
29. Mahboob A et al (2022) A Novel Construction of Substitution Box Based on Polynomial Mapped and Finite Field With Image Encryption Application. IEEE Access 10:119244–119258. https://doi.org/10.1109/ACCESS.2022.3218643
30. Manivannan D, Murugan B (2023) Image encryption using chaos based heuristic diffusion. SN Comput Sci 4(239). https://doi.org/10.1007/s42979-022-01582-3
31. Manjula G, Mohan H (2016) Constructing key dependent dynamic *S*-box for AES block cipher system 2016 2nd international conference on applied and theoretical computing and communication technology 613-617. https://doi.org/10.1109/ICATCCT.2016.7912073
32. Manzoor A, Zahid AH, Hassan MT. A New Dynamic Substitution Box for Data Security Using an Innovative Chaotic Map IEEE Access 10:74164–74174 https://doi.org/10.1109/ACCESS.2022.3184012
33. Matsui M (1994) Linear cryptanalysis method of DES cipher: Advances in Cryptology", Proceeding of the Eurocrypt'93 Lecture Notes in Computer Science 765:386–397
34. Nyberg K (1994) Differentially uniform mappings for cryptography", Proceedings of Eurocrypt'93 Lecture Notes in Computer Science 765:55–64
35. Ozturk I, Sogukpinar I. Analysis and comparison of image encryption algorithms. Int J Inf Technol 1(2):108–111
36. Lang L, Jinggen L, Ying G, Botao L (2022) A new S-box construction method meeting strict avalanche criterion. J Inf Secur Appl 66. https://doi.org/10.1016/j.jisa.2022.103135
37. Li Z et al (2020) A novel chaos-based color image encryption scheme using bit-level permutation. Symmetry 12:1497. https://doi.org/10.3390/sym12091497
38. Razaq A, Akhter S, Yousaf A, Shuaib U, Ahmad M (2022) A group theoretic construction of highly nonlinear substitution box and its applications in image encryption. Multimed Tools Appl 1–22
39. Shannon CE (1949) Communication theory of security systems The Bell Syst Tech J 28:656715
40. Sosa PM (2016) Calculating nonlinearity of Boolean functions with Walsh-Hadamard Transform UCSB, Santa Barbara 1-4
41. Teng L et al (2021) Color image encryption based on cross 2D hyperchaotic map using combined cycle shift scrambling and selecting diffusion. Nonlinear Dyn 105(2):1859–1876. https://doi.org/10.1007/s11071-021-06663-1
42. Tian Y, Lu Z (2017) Chaotic *S*-box: intertwining logistic map and bacterial foraging optimization. Math Probl Eng 2017:6969312
43. Toktas A, Erkan U (2022) 2D fully chaotic map for image encryption constructed through a quadruple-objective optimization via artificial bee colony algorithm. Neural Comput and Applic. 34:4295–4319. https://doi.org/10.1007/s00521-021-06552-z
44. Ullah A, Jamal SS, Shah T (2018) A novel scheme for image encryption using substitution box and chaotic system. Nonlinear Dyn 91(1):359–370
45. Waheed A, Subhan F, Suud MM et al (2023) An analytical review of current S-box design methodologies performance evaluation criteria and major challenges. Multimed Tools Appl. https://doi.org/10.1007/s11042-023-14910-3
46. Wang Y, Wong KW, Li C, Li Y (2012) A novel method to design *S*-box based on chaotic map and genetic algorithm. Phys Lett A 376(6–7):827–833
47. Wang Z, Bovik AC (2009) Mean squared error: love it or leave it? A new look at signalfidelity measures. IEEE Signal Process Mag 26(1):98–117. https://doi.org/10.1109/MSP.2008.930649

48. Webster AF, Tavares SE (1985) On the Design of *S*-boxes In: Williams H.C. (eds) Advances in Cryptology-CRYPTO-85 Proceedings CRYPTO 1985 Lecture notes in computer science 218 Springer Berlin Heidelberg https://doi.org/10.1007/3-540-39799-X-41
49. Wu Y, Noonan JP, Agaian S (2011) NPCR and UACI randomness tests for Image encryption. Cyber J Multidiscip J Sci Technol J Sel Areas Telecommun 1:31–38
50. Yang Y-G, Wang B-P, Pei S-K, Zhou Y-H, Shi W-M, Liao X (2021) Using M-ary decomposition and virtual bits for visually meaningful image encryption. Inf Sci 580:174–201. https://doi.org/10.1016/j.ins.2021.08.073
51. Zhang X, Gong Z (2022) Color image encryption algorithm based on 3D Zigzag transformation and view planes. Multimed Tools Appl. https://doi.org/10.1007/s11042-022-13003-x
52. Zahid AH, Iliyasu AM, Ahmad M, Shaban MMU, Arshad MJ, Alhadawi HS, Abd El-Latif AA (2021) A novel construction of dynamic S-box with high nonlinearity using heuristic evolution. IEEE Access 9:67797–67812
53. Zamli KZ, Kader A, Din F, Alhadawi HS (2021) Selective chaotic maps Tiki-Taka algorithm for the *S*-box generation and optimization. Neural Comput Appl 33:16641–16658
54. Zamli KZ (2021) Optimizing *S*-box generation based on the adaptive agent heroes and cowards algorithm. Expert Syst Appl 182:115305
55. Zhang YQ et al (2020) A new color image encryption scheme based on 2DNLCML system and genetic operations. Opt Lasers Eng 128:106040. https://doi.org/10.1016/j.optlaseng.2020.106040