RÉPUBLIQUE DU CAMEROUN
PAIX-TRAVAIL-PATRIE
****
UNIVERSITÉ DE YAOUNDÉ 1
****
CENTRE DE RECHERCHE ET FORMATION DOCTORALE EN SCIENCES, TECHNOLOGIES ET GEOSCIENCES
****
UNITE DE RECHERCHE ET FORMATION DOCTORALE SCIENCE PHYSIQUES ET APPLICATIONS
****
B.P Box 812 Yaoundé
Email : *crfd_stg@uy1.uninet.cm*

REPUBLIC OF CAMEROON
PEACE-WORK-FATHERLAND
****
UNIVERSITY OF YAOUNDÉ 1
****
POSTGRADUATE SCHOOL OF SCIENCE, TECHNOLOGY AND GEOSCIENCE
****
RESEARCH AND POSTGRADUATE TRAINING UNIT IN PHYSICS AND APPLICATION
****
B.P BOX 812 Yaoundé
Email : *crfd_stg@uy1.uninet.cm*

DÉPARTEMENT DE PHYSIQUE
DEPARTMENT OF PHYSICS

LABORATOIRE D'ÉNERGIE, DES SYSTÈMES ÉLECTRIQUES ET ÉLECTRONIQUES
LABORATORY OF ENERGY AND ELECTRICAL AND ELECTRONIC SYSTEMS

THÈME

IMPLEMENTATION D'UNE RECURRENCE DU CHAT D'ARNOLD A GRANDE PERIODE EN DIMENSION 4 SUR 2 BITS .

THÈSE

présentée en vue de l'obtention du diplôme de
DOCTORAT/Ph.D de physique

Option : Systèmes Électriques et Électroniques

Par :

GNYAMSI NKUIGWA Gaetan Gildas
Matricule : **08w0212**
Master de physique

Sous la direction de :

EYEBE Fouda Jean Sire Armand
Maître de Conférences, Université de Yaoundé I

-- 2023 --

UNIVERSITE DE YAOUNDE I
UNIVERSITY OF YAOUNDE I

FACULTE DES SCIENCES
FACULTY OF SCIENCE

DEPARTEMENT DE PHYSIQUE
DEPARTMENT OF PHYSICS

ATTESTATION DE CORRECTION DE LA THESE DE DOCTORAT/PhD

Nous, Professeurs **NJANDJOCK NOUCK Philippe, NANA NBENDJO Blaise Roméo, BODO Bertrand, NTSAMA ELOUNDOU Pascal** et Professeur **ESSIMBI ZOBO Bernard**, respectivement Examinateurs et Président du jury de la thèse de Doctorat/PhD de Monsieur **GNYAMSI NKUIGWA Gaëtan Gildas** Matricule **08W0212**, préparée sous la direction du Professeur **EYEBE FOUDA Jean Sire Armand,** intitulée : **« Implémentation d'une récurrence du chat d'Arnold à grande période en dimension 4 sur 2 bits »**, soutenue le **Vendredi, 29 Septembre 2023**, en vue de l'obtention du grade de Docteur/PhD en Physique, Spécialité **Energie, Systèmes Electriques et Electroniques**, attestons que toutes les corrections demandées par le Jury de soutenance ont été effectuées.

En foi de quoi, la présente attestation lui est délivrée pour servir et valoir ce que de droit.

Fait à Yaoundé le 19 OCT 2023

Examinateur

Professor

Le Président du Jury

ESSIMBI ZOBO Bernard
Professeur

UNIVERSITE DE YAOUNDE I * FACULTE DES SCIENCES * DEPARTEMENT DE PHYSIQUE
Le Chef de Département

Le Chef de Département de Physique

Professeur

DEDICACE

A

la famille NKUIGWA

REMERCIEMENTS

J'aimerais tout d'abord exprimer ma profonde gratitude et mes sincères remerciements à mon superviseur de thèse , Professeur EYEBE FOUDA Jean Sire Armand pour sa rigueur dans le travail et sa disponibilité tout au long de ces travaux de thèse. Durant mon parcours de cycle master et doctorat, j'ai appris de lui beaucoup de choses dans ma façon de travailler au travers de ses multiples conseils. Il ne cessait de me mettre continuellement au travail à chaque fois que je sombrait dans la facilité. En parallèle, j'ai appris de lui, que tout est possible dans la vie pour celui qui se met continuellement au travail. Je garde en idée chaque fois toutes ses orientations qu'il me donnait dans le travail et aussi il reste un modèle pour moi pour mon entreprise dans le domaine de la recherche.

Je tiens également à remercier tous les membres de jury de cette thèse qui ont pris de leur temps pour examiner ce travail afin d'apporter une plus value à la aisance de ce document.

J'aimerais exprimer mes remerciements à l'Université de Yaoundé 1, au chef de Département de Physique, Professeur NDJAKA Jean Marie Bienvenu, les enseignants, Professeur WOAFO PAUL, Professeur TCHAWOUA Paul, Professeur NANA NBENDJO,
Professeur FEWO Serges, Professeur BEN-BOLIE Germain Hulbert, Professeur ZEKENG Serges, Professeur VONDOU DERBETINI Appolinaire qui m'ont encadrés durant mon cursus académique .

Les remerciements vont en particulier aux enseignants du laboratoire d'électronique : Professeur ESSIMBI ZOBO Bernand, Professeur BODO Bertrand, Professeur MBINACK Clément et Dr. Ayissi Eyebe Guy pour leurs encouragements et soutiens.

Ma profonde gratitude va également à l'unité de prototypage des circuits du laboratoire d'électronique CPU (Circuit Prototyping Unit) situé à la facultés des sciences de L'Université de Yaoundé I où j'ai fait mes recherches, et à mon collaborateur de travail, Dr. DJEUGOUE NZEUGA Hermann, mes camarades de labo, Dr.TAGNE SAMUEL, DJEUFA GUY Morgan, WANDJA Guillene, MOUTON Mohamed, EKANI VIANNEY, CHOUAMENI Claire.

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABREVIATIONS

ACM : Arnold Cat Map

AES : Advanced Encryption Standard

ASCII : American Standard Code for Information Interchange

CBC : Cipher Block Chaining

CFB : Cipher Feedback

CPU : Central Processing Unit

DES : Data Encryption Standard

ECB : Electronic Codebook

FEAL : Fast Data Encipherment Algorithm

FPGA : Field Programmable Gate Array

IDEA : International Data Encryption Algorithm

MAC : Message Authentification Code

MATLAB : Matrix Laboratory

NIST : National Institute of Standards and Technology

NPCR : Number of Pixel Change Rate

OFB : Output Feedback

PCBC : Propagating Cipher Block Chaining

PPCM : Plus petit commun multiple

PRNG : Pseudo Random number Generator

PTSTfrFT : Phase Truncated Short Time fractional Fourier Transform

PWLCM : Piece Wise Linear Chaotic Map

QACM : Quantized Arnold Cat Map

RALM : Récurrence d'Arnold linéaire par morceau

RAM : Random Access Memory

RNG : Random Number Generator

RSA : Rivest, Shamir et Adleman

TIC : Technologie de l'information et de la communication

TRNG : True random number generator

UACI : Unified Average Changing Intensity

RESUME

Les systèmes chaotiques sont très sollicités dans la synthèse des algorithmes de cryptographie, en raison de leur ergodicité, sensibilité aux conditions initiales, et propriété de mixage. Cependant, l'usage des systèmes numériques en cryptographie moderne requiert que les trajectoires chaotiques soient digitalisées, ce qui entraine leur périodisation. Lorsqu'en plus les périodes des orbites apparemment chaotiques sont courtes comme c'est souvent le cas, le cryptogramme devient vulnérable, ne garantissant plus de ce fait la sécurité des données cryptées. De plus, l'architecture de ces systèmes dans la plupart des cas impose d'effectuer en deux étapes différentes les deux opérations fondamentales en cryptographie moderne que sont la confusion et la diffusion, ce qui contribue à augmenter le temps de chiffrement. Dans le but de limiter la dégradation de la sécurité due à la numérisation des orbites chaotiques, nous proposons dans ce travail d'implémenter la récurrence d'Arnold linéaire par morceau (RALM) en dimension 4 (4D) dont la période est grande par rapport au système d'Arnold classique. Après analyse, il en résulte que le système 4D de la RALM ainsi conçu présente une période de l'ordre de 10^8 pour une précision de 4 bits et 360 pour une précision de 2 bits. Comparé au système classique d'Arnold dont la période n'est que 3 pour une précision de 2 bits et 12 pour une précision de 4 bits, le système proposé permet d'améliorer la sécurité des cryptosystèmes tout en donnant la possibilité de les embarquer, grâce à la faible précision requise pour atteindre de grandes périodes. Contrairement aux systèmes usuels utilisant une précision de 32 bits en virgule flottante, le choix d'une faible précision dans notre approche permet de réduire le coût matériel, le temps de calcul, ainsi que la consommation d'énergie. Pour vérifier l'efficacité du système 4D codé sur 4 bits, un algorithme de cryptographie l'incluant est proposé. En effet, le système proposé combine les opérations de confusion et de diffusion en une seule étape, ce qui lui permet de gagner en temps de calcul. Cette combinaison est rendue possible grâce à

une transformation linéaire qui décompose les valeurs de pixels. La particularité d'un tel cryptosystème est son espace de clef extensible qui lui permet de prévenir les attaques futures à l'ère des ordinateurs quantiques. Lors des tests statistiques faits sur le cryptosystème, nous avons obtenu en moyenne un NPCR de 99,6078, un UACI de 33,4238 et une entropie de 7,994 en utilisant une image en niveau de gris de taille 512×512 pixels. Pour davantage réduire le coût matériel et le temps de calcul, un algorithme de cryptographie incluant la RALM en dimension 4, mais codée sur 2 bits est proposé. Le cryptosystème proposé sur 2 bits obéit au même principe de chiffrement que celui de 4 bits avec également une possibilité d'extension de l'espace des clefs, permettant ainsi de guarantir la robustesse du cryptosystème. Les tests statistiques effectués sur ce cryptosystème ont donné en moyenne, un NPCR de 99,6078, un UACI de 33,4836 et une entropie de 7.997, pour une image de 512×512 pixels, ce qui prouve sa robustesse contre les attaques connues.

Mots clés : Cryptographie, récurrence d'Arnold, systèmes dynamiques, chaos

ABSTRACT

Chaotic systems are very request for synthesis of cryptographic algorithms, due to their ergodicity, sensitivity to initial conditions, and mixing property. However, the use of digital systems in modern cryptography requires that chaotic trajectories be digitized, which leads to their periodization. When in addition the periods of the apparently chaotic orbits are short, as is often the case, the cipher becomes vulnerable, thus no longer guaranteeing the security of the encrypted data. Moreover, the architecture of those systems in most cases requires that the two fundamental operations in modern cryptography, which are confusion and diffusion, be performed in two different steps, which contributes to increasing the encryption time. In order to limit the degradation of security due to the digitization of chaotic orbits, we propose in this work to implement the piecewise wise linear chaotic map (PWLCM) in 4 dimensional (4D) whose period is large compared to the classical Arnold cat map. After analysis, it is found that the 4D PWLCM thus designed has a period of the order of 10^8 for a 4-bit precision and 360 for a 2-bit precision. Compared to the classical Arnold cat map, which has period of only 3 for 2-bit precision and 12 for 4-bit precision, the proposed system improves the security of the cryptosystems by allowing them to be embedded, due to the low precision required to achieve large periods. In contrast to conventional systems using 32-bit floating point precision, the choice of low precision in our approach reduces hardware cost, computation time, and power consumption. In order to verify the efficiency of the 4D system encoded to 4 bits, a cryptographic algorithm including it is proposed. Indeed, the proposed system combines the confusion and diffusion operations in a single step, thus saving computation time. This combination is made possible by a linear transformation that decomposes pixel values. The special feature of such cryptosystem is the extensibility of key space, which allows it to prevent future attacks in the era of quantum computers. In the statistical tests performed on the cryptosystem, we

obtained an average NPCR of 99.6078, UACI of 33.4238 and entropy of 7.994 using a grayscale image of size 512×512 pixels. Further to reduce the hardware cost and computational time, a cryptographic algorithm including the PWLCM in 4 dimensional, but encoded on 2 bits is proposed. The proposed 2-bit cryptosystem obeys the same encryption principle as the 4-bit, with the possibility to extend the key space, allowing to guarantee robustness of the cryptosystem. Statistical tests carried out on this cryptosystem gave, on average NPCR of 99.6078, UACI of 33.4836 and entropy of 7.997, for image of 512×512 pixels, which proves its robustness against known attacks.

Keywords : Cryptography, Arnold cat map, dynamic systems, chaos

INTRODUCTION GENERALE

L'évolution des technologies de l'information et de la communication (TIC) a fait qu' Internet est devenu un outil majeur d'échange d'informations entre individus, les agences gouvernementales, les entreprises, les institutions académiques, les institutions militaires. [8]. D'après le site www.e-works.fr, on dénombre près de 4,39 milliards d'internautes à travers le monde en 2019. Toutefois, ces informations échangées sur la toile sont en grande partie confrontées à des attaques informatiques. Dans le souci d'intégrité, d'authenticité et de confidentialité de celles-ci, l'on fait recours à la cryptographie. En effet, la cryptographie est une science qui consiste à protéger l'information en la transformant en un format illisible appelé cryptogramme. Parmi les informations échangées à travers les réseaux de communication, les images numériques occupent une place de choix, car leur transmission et leur stockage font l'objet d'un grand nombre d'utilisation en ce 21e siècle. .

Les algorithmes de chiffrement d'images basé sur le chaos ont fait l'objet de nombreux travaux de recherches durant cette dernière décennie [9, 10, 11]. Toutefois, ces systèmes chaotiques présentent d'excellentes propriétés intrinsèques telles que : la sensibilité aux conditions initiales, les dynamiques de type aléatoire, l'imprédictibilité, conduisant ainsi à les utiliser pour des applications cryptographiques. Parmi ces systèmes, ceux les plus utilisés sont : la récurrence logistique 2D [12], la récurrence d'Henon 2D [13], la récurrence de Baker 3D [14], la récurrence d'Arnold [15] etc.

En effet, Zhu et al [9] proposent un cryptosystème en utilisant la récurrence du chat d'Arnold pour effectuer la permutation des bits des pixels d'une image et la récurrence logistique pour la diffusion. Dans la meme optique, Ye et al [10] propose un cryptosystème en s'inspirant cette fois ci de la récurrence d'Arnold généralisée pour améliorer la sécurité du cryptosystème

définit précédemment. Dans la même lignée, Guan et al. [16] propose un cryptosystème robuste en utilisant la récurrence d'Arnold pour la permutation des pixels, puis la récurrence de Chen pour le processus de diffusion. Pour accroitre d'avantage le niveau de sécurité, une combinaison linéaire de la carte chaotique linéaire par morceaux et l'équation linéaire de Diophantine (LDE) est proposée pour le chiffrement d'images [17]. Bien que l'image chiffrée fut obtenu après un seul tour, l'inconvénient est que le processus de chiffrement de l'équation linéaire de Diaphantine (LDE) produit des nombres réels, ce qui conduit à une conversion de ces nombres en nombres entiers pour effectuer les opérations de chiffrement. Cette conversion de données entraine souvent dans la majeur partie des cas, l'allongement du temps de calcul du cryptosystème.

Cependant, la récurrence du chat d'Arnold présenté dans ces cryptosystèmes, a un mauvais effet sur le chiffrement des images car elle est toujours périodique ce qui entraine la faiblesse de ceux-ci. C'est ainsi que, de nombreux travaux de recherches se sont intéressés à ce problème, mais seuls Dyson et Falk [18] ont pu trouver une relation de récurrence permettant d'évaluer ces périodes dans un espace de phase discret. Il convient de relever que cette période reste toujours faible pour différentes combinaisons de paramètres de contrôles choisis et conditions initiales . Toutefois l'implémentation de ce système dans les cryptosystèmes, conduit souvent à des opérations de confusion et de diffusion en deux étapes distinctes qui impliquent l'allongement du temps de calcul.

En effet, la récurrence d'Arnold est un système chaotique en dimension 2 qui fut conçu en 1960 par le Russe VLADIMIR ARNOLD, qui a démontré son application en brouillant ou en permutant les pixels de l'image d'un chat et après un certain nombre d'itérations appelé période, il parvenait à retrouver l'image de départ. Comparé à d'autres systèmes dynamiques, la récurrence d'Arnold présente la particularité d'être simple à réaliser et à embarquer dans un microcontrôleur. De plus, elle intervient dans la majeur partie des cas, pendant la phase de permutation des pixels dans le processus de chiffrement d'images.

Toutefois, la faiblesse de la période de la récurrence du chat d'Arnold dans les cryptosystèmes, conduit ainsi la vulnérabilité de ceux-ci contre les différentes attaques connues. Dès lors pour palier à cette situation, nous proposons dans ce travail, une implémentation de la récurrence d'Arnold en dimension 4 (4D), codée sur 2 bits ayant une grande période.

Pour concevoir un tel système, nous proposons d'abord une modification de la récurrence d'Arnold classique, en ajoutant sur chaque variable d'état des termes de pertubations non linéaires. Le système obtenu après cette modification est appelé la récurrence d'Arnold linéaire par morceau (RALM) en dimension 2 (2D). Ainsi, par la suite, pour les besoins d'utilisation

cryptographique, nous proposons une extension 4D de la RALM codée respectivement sur 4 bits et 2 bits . Pour vérifier l'efficacité de ces systèmes, nous l'avions implémenté dans les cryptosystèmes respectivement sur 4 bits et 2 bits. Les résultats et les tests d'analyses statistiques effectués sur ces cryptosystèmes ont prouvé qu'ils sont efficaces pour des applications cryptographiques. De plus, le système proposé RALM 4D présente une période suffisamment grande permettant ainsi aux cryptosystèmes respectivement sur 2 bits et 4 bits proposés, de résister contre les attaques à forces brutes.

Ce travail de thèse est un atout pour améliorer les problèmes d'insécurité lors d'échange d'information via des réseaux non sécurisé (Intranet, Internet, Wifi, etc). C'est également nécessaire de l'utiliser pour la confidentialité, l'intégrité et l'authenticité d'une information d'ordre sécrète, par exemple les résultats d'un diagnostic médical prescrit par un médecin, une information d'enquête militaire, des transactions bancaires via l'utilisation des comptes bancaires, etc. L'intérêt d'implémenter ces cryptosystèmes sur des faibles précisions (4 bits et 2 bits), contribue à l'intégration de ceux-ci dans un capteur électronique (détecteur de présence, caméra de surveillance, etc.), une puce électronique (circuit intégré, microcontroleur, etc.) .

Le document de thèse présenté est organisé comme suivant :

- Au chapitre 1, nous présenterons l'état de l'art, en faisant une revue générale de quelques systèmes dynamiques chaotiques en présentant également les applications de ceux-ci en physique et dans la société.
- Au chapitre 2, nous présentons les méthodes et les outils utilisés pour l'implémentation du cryptosystème sur 2 bits en utilisant la RALM 4D.
- Au chapitre 3, nous présentons en détail les résultats des algorithmes de chiffrement proposés, d'une part les résultats obtenus sur une précision de 4 bits, et d'autre part les résultats obtenus sur 2 bits et enfin quelques discussions .

CHAPITRE 1

GENERALITES SUR LES SYSTEMES CHAOTIQUES

Introduction

A la base, les systèmes chaotiques, sont des fonctions qui s'éxécutent de façon itérative à partir d'une condition initiale donnée pour obtenir une serie de valeurs entières ou réelles. De ces systèmes, se dégage la notion du chaos qui est un phénomène naturel qui avait été découvert en 1963 par le météorologiste américain Edward lorenz . Depuis une décenie, il existe une relation étroite entre le chaos et le domaine de la cryptographie. Les propriétés intrinsèques de ces systèmes, possèdent leur correspondance dans la modélisation de cryptosystèmes traditionnels tels que, DES (Data Encryption Standard), RSA (Rivest, Shamir et Adleman) et IDEA (International Data Encryption Standard). Parmi ces propriétés, nous avons : l'ergodicité, la sensibilité aux conditions initiales où paramètres de controles, le mixage, le déterminisme et la complexité [19]. Ainsi, les techniques de cryptographie basées sur le chaos, sont considérées comme bonnes pour une utilisation pratique, car elles fournissent une bonne qualité de chiffrement [20]. Ces techniques ont été présentées dans la littérature en utilisant différents algorithmes de cryptographie [21, 22, 23]. Dans ce chapitre, nous allons tout d'abord présenter quelques notions générales sur le chaos en explicitant quelques systèmes chaotiques, ensuite on présentera les outils d'analyse de ces systèmes et enfin les applications de ceux-ci en cryptographie .

1.1) Notions sur la théorie du chaos

Le terme chaos dérive du mot grec "Kαoc" qui signifie imprévisibilité. Cependant, depuis une vingtaine d'années, on attribue le terme chaos à des phénomènes qu'on ne peut à priori déterminer une logique, et qui sont régis par des équations linéaires ou non. Parmi ces phénomènes, nous pouvons citer : les variations météorologiques, les arythmies cardiaques, les oscillations du cerveau, le mouvement des étoiles etc. Ces équations ou systèmes d'équations sont dites déterministes. En général ils présentent parfois un caractère complexe , non périodique [24]. En général, ces systèmes possèdent quelques propriétés tels que : la sensibilité aux conditions initiales, le caractère aléatoire et l'ergodicité. Une légère variation des conditions initiales, conduit à une grande variation des valeurs générées par le système. La distribution des variables d'états s'effectue uniformément dans l'espace des phases. Ainsi, ces propriétés du chaos conduisent à l'utiliser dans le domaine de la cryptographie [25, 26]. Dans la littérature, plusieurs systèmes chaotiques ont été utilisés dans le processus de cryptage d'images et les plus utilisés sont : la récurrence logistique, la récurrence d'Hénon, de Tent, de Sine, d'Arnold etc.

1.1.1) La récurrence logistique

La récurrence logistique fût utilisée pour la première fois en 1845 par P.F Verhust pour modéliser le développement d'une population dans un environnement bien déterminé [27]. En effet, il s'agit d'une équation non linéaire à une dimension (1-D) continu dans l'espace et discret dans le temps donnée par :

$$x_{n+1} = \mu \cdot x_n(1 - x_n) \tag{1.1}$$

où $x_n \in [0,1], n \in \mathbb{N}$, x_0 représente la population initiale avant son développement et μ est un paramètre de contrôle pris dans l'intervalle $[0,4]$. L'évolution de la population au cours du temps peut être représenté par le diagramme de bifurcation donnée à la figure 1 ci-dessous.

D'après la figure, nous remarquons que le système devient chaotique lorsque $\mu \in [3.5,4]$. La récurrence logistique présente quelques caractéristiques importantes telles que, la grande sensibilité aux conditions initiales et paramètres de contrôles, la non- linéarité, ce qui fait de lui un outil pour le chiffrement des données. Toutefois, il présente certaines faiblesses, tel qu'un faible espace des valeurs générées par la récurrence. Cette faiblesse, conduit à ne pas l'utiliser à elle seule pour concevoir un algorithme de chiffrement. C'est pour cela qu'on emploie d'autres récurrences tel que la récurrence d'Henon, d'Arnold pour accroitre le degré d'aléa du

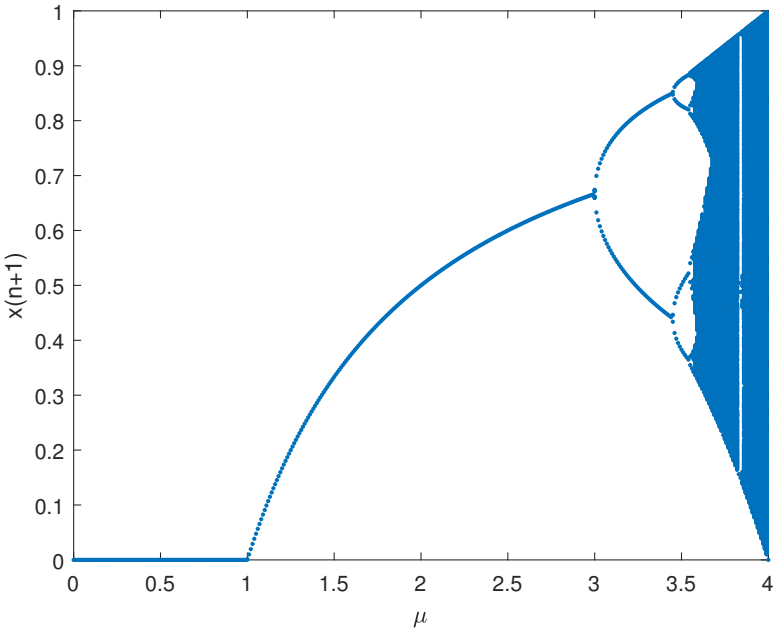


FIGURE 1.1 – Diagramme de bifurcation

cryptosystème.

1.1.2) La récurrence d'Henon

La récurrence d'Henon est un système dynamique, utilisée pour la première fois en 1969 par Michel Henon. En effet, il s'agit d'un système d'équation discret en temps et continu en espace qui obéit aux critères de choix d'un système chaotique [28]. Le système d'Henon est définit par :

$$\begin{cases} x_{n+1} = y_n + 1 - ax_n^2 \\ y_{n+1} = bx_n \end{cases} \tag{1.2}$$

Les variables, x et y sont les valeurs itérées de la récurrence, où, $n = 0, 1, 2, ...$, sont les indices d'itérations. Ce système dépend de 2 paramètres (paramètres de contrôles) réels, données par a et b. On observe, d'après la figure (1.2) présentée ci-dessous, pour $a = 1.4$ et $b = 0.3$, le système dynamique présente un caractère chaotique. Nous pouvons aussi remarquer que les valeurs itérées en indices (le temps) donnent les valeurs discrètes (les valeurs entières) tandis que les variables x et y obtenues sont continues (valeurs réelles) [24] .

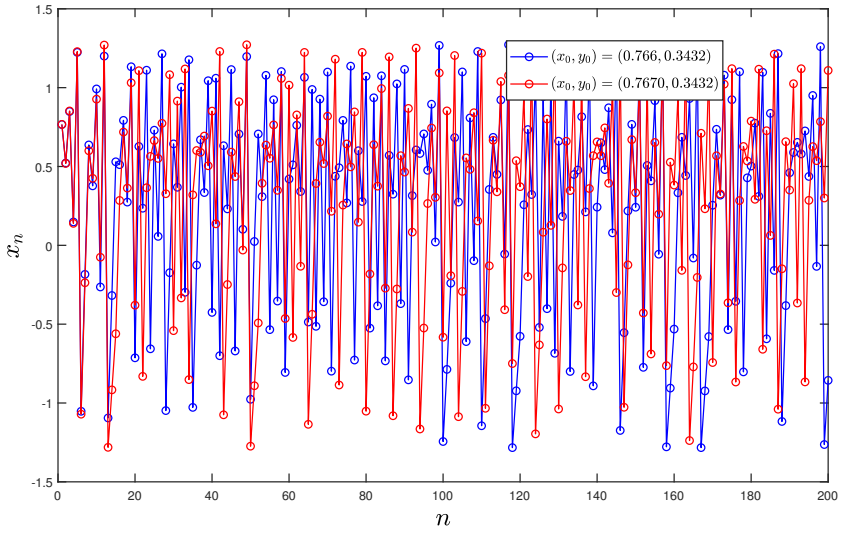


FIGURE 1.2 – Evolution temporelle de x pour deux paires de conditions initiales légèrement différentes ($a = 1.4$, $b = 0.3$, $(x_{01}, y_{01}) = (0.766, 0.3432)$ et $(x_{02}, y_{02}) = (0.767, 0.3432)$)

1.1.3) *La récurrence de Tent*

Il s'agit d'une carte linéaire, continue, et définit par morceaux, possédant un maximum unique, dans la région chaotique, en terme de densité invariante et de spectre de puissance [29].

La récurrence chaotique de Tent [29] est donnée par :

$$x_{i+1} = f(x_i, \mu) \tag{1.3}$$

$$f(x_i, \mu) = \begin{cases} \mu x_i \quad \text{if} \quad x_i < 0.5 \\ \\ \mu(1 - x_i) \quad \text{ailleurs} \end{cases} \tag{1.4}$$

Les valeurs réelles $x_i \in [0,1]$, avec $i \geq 0$. C'est une fonction définit de $[0,1]$ vers $[0,1]$ et possédant un paramètre de contrôle $\mu \in [0,2]$. L'ensemble des valeurs $\{x_0, x_1, ..., x_n\}$ est appelé orbite du système où x_0 est la condition initiale. Il s'agit d'un système à 1-D, possédant les propriétés chaotiques similaires à la récurrence logistique [30]. Ce système présente les exposants de Lyapounov dans un intervalle unité avec comme paramètres de contrôles $\mu \in \{0,2\}$. La représentation de l'évolution de la fonction f au cours du temps (le nombre d'itérations) de la récurrence de Tent peut être représenté à travers la figure (1.3) ci-dessous.

Comme nous pouvons le remarquer d'après la figure ci-dessus, la récurrence est fortement

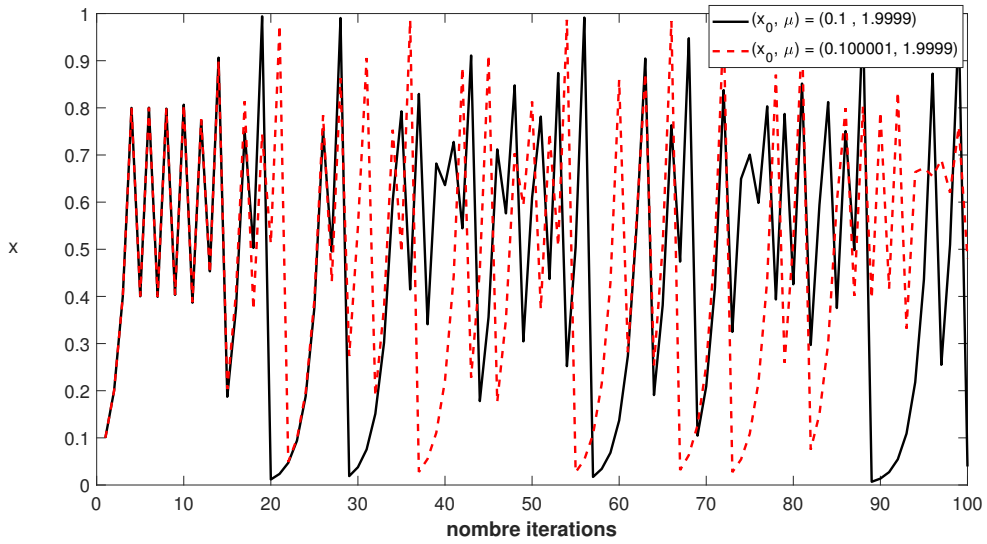


FIGURE 1.3 – Série chronologique de la séquence de Tent générée pour 2 conditions initiales légèrement différentes $|\bar{x_0} - x_0| = 10^{-6}$. En trait fort couleur noir correspond $(x_0, \mu) = (0.1, 1.9999)$ et en trait interrompu rouge $(x_0, \mu) = (0.100001, 1.9999)$

sensible aux conditions initiales, présentant des trajectoires différentes due à une faible variation de la condition initiale. Cet observation est un outil majeur pour caractériser que le comportement de la récurrence de Tent est dite chaotique.

1.1.4) La récurrence de Sine

C'est un système chaotique en dimension 1, dont les valeurs en sortie sont comprises dans le domaine $[0, 1]$. C'est également une équation qui obéit aux memes propriétés chaotiques que la recurrence logistique. Son expression mathématique est donnée par [31] :

$$x_{i+1} = \eta sin(\pi x_i) \tag{1.5}$$

$-1 < x_i < 1$. Le paramètre de contrôle est $\eta \in [0, 1]$. Le système présente un caractère chaotique lorsque, $\eta \in [0.87, 1]$.

Nous pouvons le voir à travers le diagramme de bifurcation présenté ci-dessous.

Nous pouvons aussi noter que, le diagramme de bifurcation de la récurrence de Sine est similaire à celui de la récurrence logistique présenté à la section (1.1).

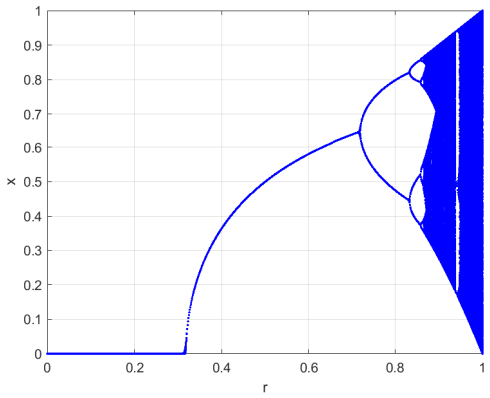


FIGURE 1.4 – Diagramme bifurcation récurrence de Sine

1.1.5) La récurrence d'Arnold

La récurrence d'Arnold a été étudiée pour la première fois par le mathématicien russe, Vladimir Arnold pour brouiller l'image d'un chat d'où son nom donnée dans la littérature appelé récurrence du chat d'Arnold. C'est un système d'équation qui possède des propriétés très intéressantes,tels que, l'ergodicité, la sensibilité aux conditions initiales, le mixage, le déterminisme,la conservation en terme de surface et l'irréversibilité. Son expression mathématique est donnée par :

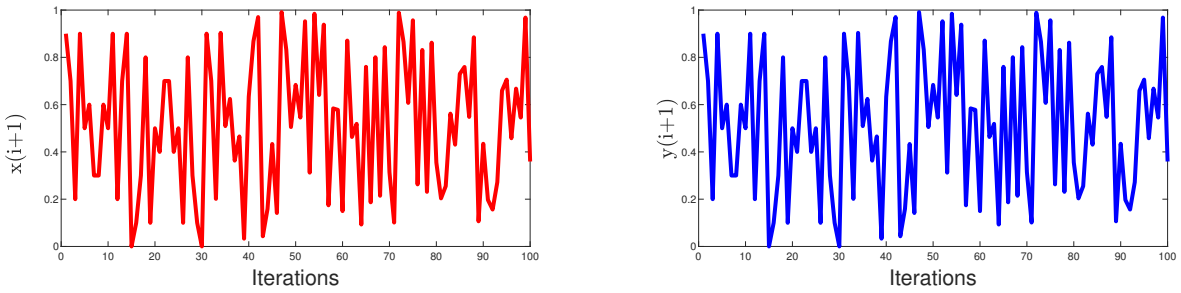
$$\begin{pmatrix} x(n+1) \\ y(n+1) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} x(n) \\ y(n) \end{pmatrix} \bmod 1 \tag{1.6}$$

où, $x(n), y(n) \in [0,1]$. Pour les besoins d'utilisations dans les applications diverses tels que, la stéganographie [32], la cryptographie [33], les générateurs de nombres aléatoires , il est nécessaire de la mettre sous la forme discrète. La forme discrétisée de l'équation (1.6) est donnée par :

$$\begin{pmatrix} x(n+1) \\ y(n+1) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} x(n) \\ y(n) \end{pmatrix} \bmod N \tag{1.7}$$

où, $x(n), y(n) \in \{0,1,...,N-1\}$, avec $N \in \mathbb{N}^*$ et N peut s'exprimer en fonction de la précision de calcul de notre calculateur qui représente en d'autre termes le nombre de bits n, par : $N = 2^n$. Le système discret d'Arnold présenté ci-dessous transforme une image en permutant ses pixels de façon aléatoire. Cependant, si l'on itère la récurrence d'Arnold un nombre de fois d'itérations, l'image originale finit par réapparaitre. Le nombre d'itérations considéré est connu sous le nom de la période. Cette période dépend de la taille de l'image prise à l'entrée [34]. Les courbes re-

présentant l'évolution des séquences x et y au cours du temps de la récurrence d'Arnold (1.6) peuvent être données comme la figure ci-dessous. Nous observons que les séquences x et y



(a) (b)

FIGURE 1.5 – Evolution continue de la récurrence d'Arnord avec $x_0 = 0.9$ et $y_0 = 0.9$: (a) cas de la séquence x , (b) cas séquence y

génèrent des valeurs pseudo-aléatoires après 100 itérations du système , ce qui implique que les séquences obtenues sont imprévisibles. Ce qui conduit donc, à prouver qu'il s'agit lui aussi d'un système chaotique. Cette récurrence en particulier fera l'objet de notre étude de thèse. Les détails y afférents seront détaillés dans les chapitres 2 et 3.

1.2) Analyse des systèmes dynamiques

Dans l'étude des systèmes dynamiques continus ou discrets, il est important voir nécessaire d'effectuer des analyses pour étudier globalement le comportement où le caractère de ceux-ci au cours du temps . De manière générale, pour effectuer une étude analytique de tels systèmes, il existe plusieurs outils . Ces outils peuvent être regroupés en 2 classes : les outils qualitatifs et les outils quantitatifs.

1.2.1) Outils qualitatifs

a) Le diagramme de bifurcation

En mathématique, et en particulier dans l'étude des systèmes dynamiques, un diagramme de bifurcation illustre les valeurs visitées asymptotiquement (points fixes, points périodiques, attracteurs chaotiques) par un système en fonction d'un paramètre [35]. La bifurcation renvoie à l'étude du comportement d'un système lorsqu'on fait varier certains de ses paramètres de contrôles. Autrement dit, il s'agit en d'autre termes d'étudier l'instabilité du système due à la modification d'un de ses paramètres de contrôles. Nous pouvons l'observer sur la figure (1.6) donnée ci-dessous. Elle représente le diagramme de bifurcation de la récurrence d'Henon en

faisant varier le paramètre de contrôle a. On déduit que le système est chaotique dans la plage $\{0.6, ..., 1.5\}$.

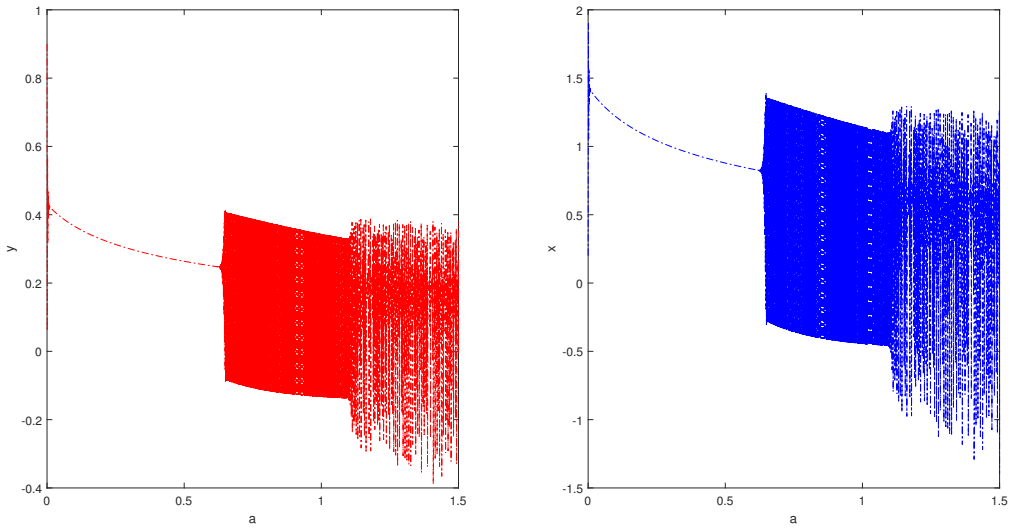


FIGURE 1.6 – Diagramme de bifurcation récurrence d'Henon en x(en bleu) et y(en rouge) pour $b = 0.3$, en faisant varier a

b) Section de Poincaré

La technique d'analyse des solutions d'équations différentielles, développée par Poincaré au début du 20e siècle, a pris une grande importance dans l'étude moderne des systèmes dynamiques. La section de Poincaré est un outil graphique qui facilite l'étude des systèmes dynamiques en ramenant l'analyse d'un système différentiel à temps continu de dimension m à celle d'une application à temps discret de dimension $m-1$.

Dans le cas d'un système dynamique de dimension 3, la section de Poincaré se dessine sur une surface (Σ) appartenant au plan $(q_i, q_j)_{i \neq j}, i, j = 1,2,3$ avec q_i, $i = 1,2,3$ les coordonnées généralisées du système.

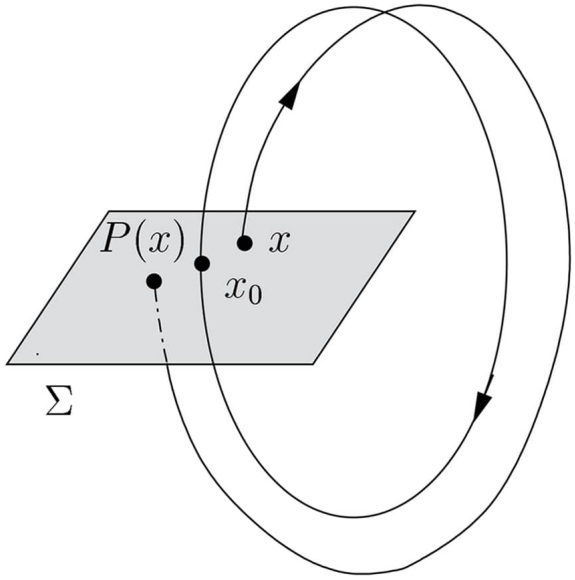


FIGURE 1.7 – Section de Poincaré : la trajectoire de phase coupe le plan (Σ)

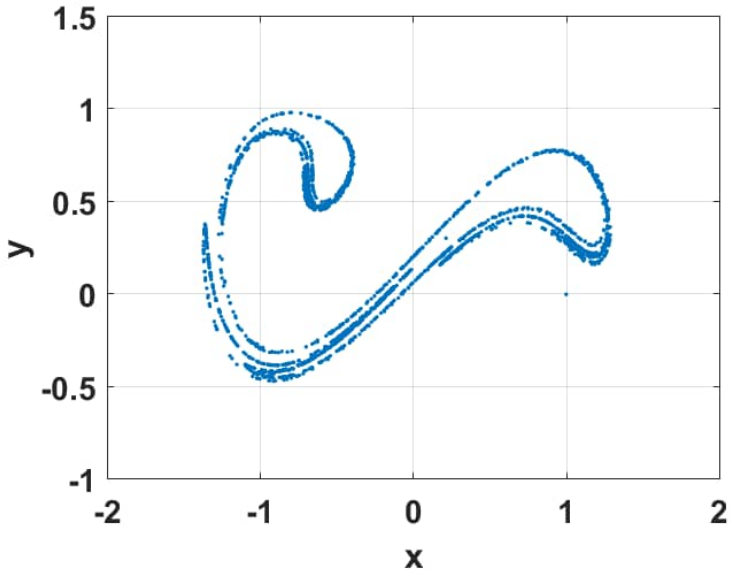


FIGURE 1.8 – Section de Poincaré de l'oscillateur de Duffing

La section de Poincaré est représentée à la figure (1.7) par l'ensemble des points d'intersection P(x) de la trajectoire de phase avec la surface (Σ). Pour une meilleure analyse qualitative, la surface Σ doit etre choisie de manière à savoir le plus grand nombre de points d'intersections comme celle choisie à la figure (1.8) pour l'oscillateur de Duffing [36].

c) Portrait de phase

Un système dynamique est en général caractérisé par un système d'équations continus ou discrètes. Le système est constitué par des variables spatiales dites variables d'états qui

donnent les différentes positions occupées par une particule (un échantillon, une donnée,etc.) au cours du temps. L'ensemble de toutes les positions possibles occupées par ces particules constitue le portrait de phase. C'est un outil important dans l'analyse de systèmes dynamiques car celui-ci permet d'observer les différents attracteurs dans un système, de justifier le caractère aléatoire ou chaotique d'un système. Les positions occupées peuvent former des attracteurs étranges comme : l'attracteur d'Henon, de Lorenz , de Duffing, etc. Le portrait de phase représente la répartition des variables d'états du système dans le plan où dans l'espace . Nous avons à la figure ci-contre les portraits de phases de quelques systèmes. D'après l'observation, on déduit que tous les systèmes dynamiques ne forment pas d'attracteurs. Dans le cas de la figure (1.10), la récurrence d'Arnold ne présente pas d'attracteurs . La distribution des valeurs est repartie uniformément dans le plan. Les systèmes qui nous ont permis d'obtenir les portraits de phases présentés ci-dessous, sont définis par :

* Récurrence d'Henon

$$\begin{cases} x_{n+1} = y_n + 1 - a x_n^2 \\ y_{n+1} = b x_n \end{cases} \tag{1.8}$$

où, a et b, les paramètres de contrôles de la récurrence choisis tels que : $a = 1.4$ et $b = 0.3$.

* Système de Lorenz

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases} \tag{1.9}$$

où, $\sigma = 16.0$; $\rho = 45.92$ et $\beta = 4.0$ sont les paramètres de contrôles du système [37].

* Récurrence d'Arnold

$$\begin{cases} x(t+1) = x(t) + \alpha y(t) \\ y(t+1) = \beta x(t+1) + y(t) \end{cases} \text{ mod } 1 \tag{1.10}$$

où, α et β sont les paramètres de contrôles de la récurrence tels que : α et $\beta \in [0, 1)$

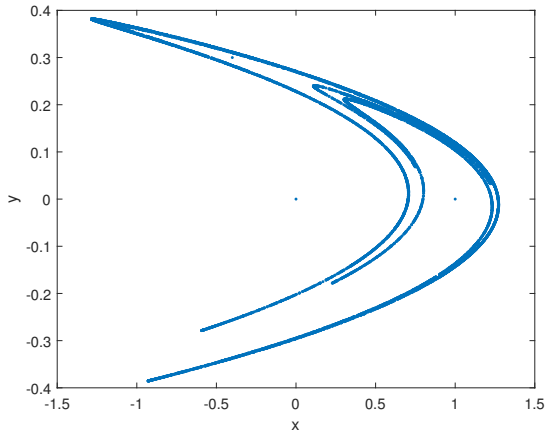
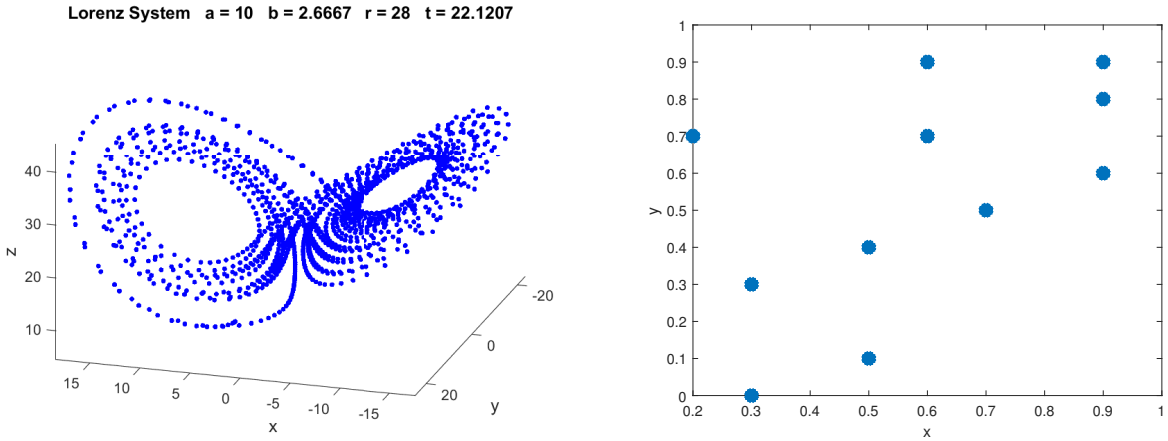


FIGURE 1.9 – Portrait de phase récurrence Henon [1]



(a) (b)

FIGURE 1.10 – Portrait de phase : (a) système de Lorenz [2], (b) Recurrence d'Arnold

1.2.2) Outils quantitatifs

a) Test 0-1

C'est une méthode quantitative d'analyse de systèmes dynamiques donnant deux valeurs numériques soit 0 ou 1 permettant de valider si le système dynamique est chaotique où pas. Soit un système dynamique caractérisé par la variable d'état $x(t) = [x_1(t), x_2(t), x_3(t), ..., x_n(t)]$. Soit l'observable donnée par : $\Phi(t) = \Phi(x(t))$. Le test s'effectue en faisant les équations suivantes [38] :

$$\theta(t) = ct + \int_0^t \Phi(x(s))ds \tag{1.11}$$

où c est une constante prise de façon arbitraire.

$$p(t) = \int_0^t \Phi(x(s))cos(\theta(s))ds \tag{1.12}$$

$$M(t) = \lim_{T\to\infty} \frac{1}{T}\int_0^T [p(t+\tau) - p(\tau)]^2 d\tau \tag{1.13}$$

$$K = \lim_{t\to\infty} \log \frac{M(t)}{t} \tag{1.14}$$

Lorsque $K = 0$, la trajectoire effectuée est dite régulière, donc le système est non chaotique ; dans le cas où $K = 1$, le système présente un caractère chaotique.

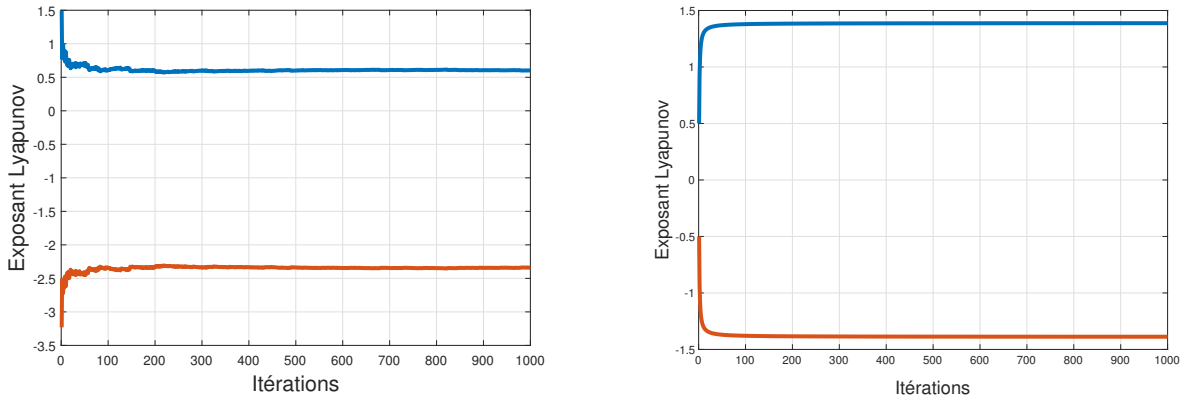
b) Exposant de Lyapunov

Les exposants de Lyapunov sont les taux moyens de convergence où de divergence calculés de façon exponentielle, des orbites proches dans l'espace de phase. Ils consistent à déterminer les axes principaux d'une sphère en utilisant les conditions initiales. Ces axes sont definis par l'évolution dans le temps des équations du système dynamique. Tout système dynamique ayant au moins un exposant de Lyapunov positif est considéré chaotique. L'amplitude de l'exposant donne l'échelle de temps pour lequel le système dynamique devient imprévisible [39] .
Tout système dynamique continu n'ayant pas de points fixes aura au moins un exposant qui vaut 0 [40]. L'exposant de Lyapunov est calculé pour chaque dimension et il dépend de la longueur de l'axe principal de l'ellipsoïde. Il peut être calculé en utilisant la relation suivante :

$$\lambda_i = \lim_{t\to+\infty} \frac{1}{t} \log_2 \frac{p_i(t)}{p_i(0)} \tag{1.15}$$

où $p_i(t)$ représente la longueur de l'axe principal de l'ellipsoïde définit à un temps t quelconque et $p_i(0)$ représente la longueur de cet axe à l'instant initial c'est à dire à $t = 0$ [41].
Les exposants nous donnent une idée de la contraction ou de l'expansion d'une direction spécifique dans l'espace des phases.
Les courbes donnant les dynamiques des exposants de la récurrence d'henon et d'Arnold, peuvent être visualisées à travers les courbes ci-dessous.

D'après les observations faites sur ces courbes , nous pouvons déduire que ces récurrences présentent un caractère chaotique car on a au moins un exposant positif , la récurrence du chat d'Arnold est de plus conservatrice car la somme de ses exposants valent 0 et qui n'est pas le cas



(a) (b)

FIGURE 1.11 – Diagramme des exposants de Lyapunov : (a) Henon , (b) Chat d'Arnold

de la récurrence d'Henon. Cette observation, nous conduit une fois de plus l'intérêt d'utiliser la récurrence du chat d'Arnold pour la cryptographie des images. Bien que ces systèmes dynamiques présentent un atout majeur pour l'analyse, il convient d'explorer quelques applications de ceux-ci dans la physique.

c) Entropie de permutation

Pour quantifier la complexité d'un système dynamique, diverses méthodes ont été développées parmi lesquelles les entropies [42, 43, 44], les exposants de Lyapunov, la dimension de corrélation [45]. Toutefois, ces méthodes ne sont pas appropriées quand il s'agit de mesurer la dynamique d'une série arbitraire. Elles ne tiennent pas compte de l'ordre temporel des données, leur traitement numérique requiert beaucoup de temps et sont inefficaces en présence du bruit, quoique de récentes recherches telles que la complexité algorithmique de Kolmogorov-Chaitin essaient d'apporter une solution à ce problème [46].

C'est ainsi, en développant les recherches autour de la dynamique symbolique des systèmes [47, 48, 49], Bandt et al. ont combiné ces concepts avec celui de l'entropie pour créer leur nouvelle approche de mesure de la complexité appelée, entropie de permutation [50]. Elle se base sur les relations d'ordre entre les valeurs des motifs de taille prédéfinie d'une série temporelle arbitraire donnée. Elle consiste en effet à repartir les données d'une séquence en partitions de taille d (qu'on appellera plus tard dimension d'intégration) et à ressortir le motif ordinal qui découle de chaque partition. L'entropie de cette série est alors donnée par la relation :

$$H = -\sum_{i=1}^{n!} p_j \log_2 p_j \tag{1.16}$$

où, p_j représente les fréquences relatives d chaque partitions et $n!$, le nombre total de partitions possibles.

1.3) Applications des systèmes chaotiques

1.3.1) Générateur de nombres pseudo- aléatoires (PRNG)

Le premier générateur de nombres pseudo-aléatoires fut proposé en 1946 par John von Neumann [51]. La séquence produite par celui-ci était de mauvaise qualité mais reste l'un des principaux jalons dans l'évolution des générateurs pseudo-aléatoires. Un générateur de nombres pseudo-aléatoires est une méthode déterministe permettant de produire à partir des valeurs prises en entrées de façon aléatoire (appelées graine), des valeurs plus importantes en sortie ayant aussi l'apparence aléatoire. De ce fait, la graine définie à l'entrée possède une complexité finie, ainsi le générateur de nombres pseudo-aléatoire (PRNG) pourrait être facilement attaqué si les valeurs issues de celle-ci (la graine) ne sont pas aléatoire. Le générateur de nombres pseudo-aléatoire (PRNG) est un outil beaucoup utilisé dans le domaine de la sécurité informatique et nécessite donc de produire des séquences aléatoires pour accroitre cette sécurité [52].

Supposons qu'un générateur de nombres pseudo-aléatoire est définit par le groupe suivant : $\zeta = (V, F, Y, G)$; $V = \{X_i : i = 1, 2, ..., q\}$ représente les différents états du système.

$F : V \to V$ est la fonction d'opération où le système dynamique .

$Y = \{Y_i : i = 1, 2, ..., q\}$ est l'ensemble des valeurs produites en sorties issues du générateur.

$G = V \to Y$ représente la fonction permettant de générer les valeurs de sortie. Les valeurs initiales (la graine) sont données par : $X_i(0) = \{x_{ij}(0) : j = 1, 2...m\}$; Les valeurs en sorties sont définies par : $Y_i(n) = \{y_{ij}(n) : j = 1, 2...m\}$; Avec, $y_{ij}(n)$ représentant les valeurs binaires (0 où 1) ; Les fonctions F et G sont définies par les relations suivantes [53] :

$$X_i(n) = F(X_i(n-1)) \tag{1.17}$$

$$Y_i(n) = G(X_i(n)) \tag{1.18}$$

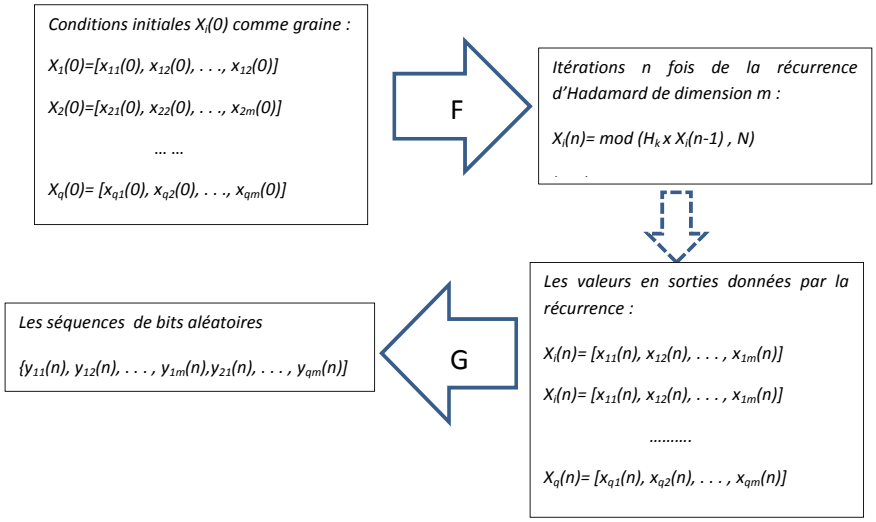


FIGURE 1.12 – Diagramme bloc du générateur de nombres aléatoires (PRNG) utilisant la récurrence d'Hadamard de dimension m (m une valeur fixée

Cang et al. [54] ont conçu un générateur en utilisant le système de Sprott. Ils ont utilisé le calcul de la période de précision finie (FPPC) pour évaluer le taux de répétition de la séquence. Pour valider leur générateur, ils ont utilisé le test statistique de NIST.

Lv et al. [55] ont présenté les différents défauts dans la programmation en C des générateurs de nombres pseudo-aléatoires. Ainsi, ils ont proposé une méthode pour améliorer leur faisabilité.

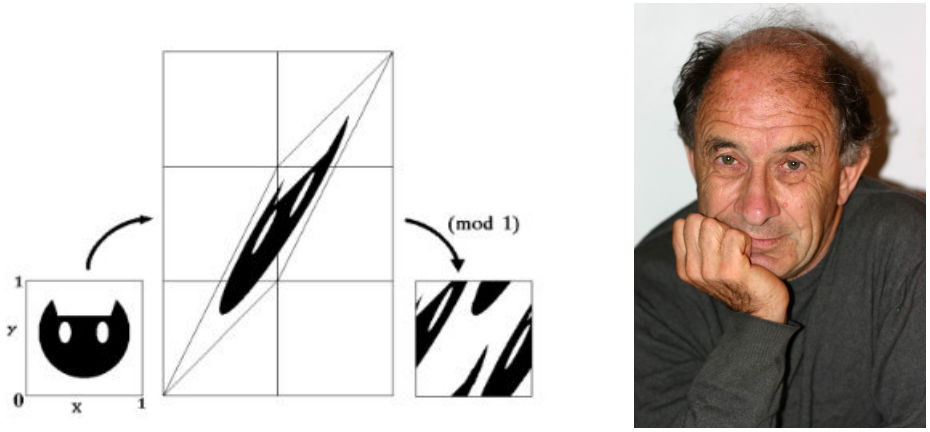
Stoyanov et al. [56] ont fait l'implémentation sur Arduino d'un PRNG en utilisant la récurrence de Ikeda. L'expression mathématique de la récurrence de Ikeda est donnée par :

$$z_{n+1} = A + Bz_n e^{i(|z_n|)^2+C} \tag{1.19}$$

Ils ont observé que l'entropie donnait 7.999 bits/octet. Ils ont testé le système également pour une taille de données de 250 millions d'octets et observer que le système passait le test de NIST. Le coefficient de corrélation valait 0.0006.

Yu et al.[57] proposent un système memristif hyperchaotique en 5-D pour générer les nombres pseudo-aléatoires.Ils ont utilisé la carte FPGA, pour l'implémentation. Les simulations ont été effectuées sur VIVADO-2018.3 et la synthèse sur ZYNQ-XC7Z020. Le générateur proposé peut atteindre une fréquence de 138.33 MHz avec comme débit de 15.367 Mb/sec.

Avarouglu et al.[58] ont développé le générateur en utilisant la récurrence du chat d'Arnold. Le coefficient de corrélation obtenu était faible environ 0.06. Cette récurrence effectue les opérations de transformation de l'image d'un chat en faisant des permutations sur celle-ci afin d'obtenir une image brouillée [58].



(a) (b)

FIGURE 1.13 – Récurrence du chat d'Arnold : (a) Application , (b) Vladimir Arnold

Djeugoue et al. [59] présentent la récurrence linéaire par morceaux en 2-D en utilisant la récurrence du chat d'Arnold . Pour générer les séquences aléatoires, on apporte les modifications sur chaque variable d'état de la récurrence d'Arnold. Les valeurs produites ont un caractère pseudo-aléatoire, car pour une précision de 10 bits, on a relevé une période de 1.09×10^{513}. L'implémentation a été faite sur la carte Zynq 7020 FPGA. Le test de NIST a été effectué pour prouver l'efficacité de ce générateur.

1.3.2) Vrai Générateur de nombres aléatoires (TRNG)

Les nombres aléatoires sont nécessaires dans de nombreux domaines : cryptographie, calcul et simulation de Monte-Carlo, essais et étiquetage industriels, jeux de hasard, etc. Nous sommes partis du principe que les nombres aléatoires ne peuvent pas être calculés comme les ordinateurs qui fonctionnent de manière déterministe, car ils ne produisent pas les nombres aléatoires.

Contrairement aux PRNG, les vrais générateurs de nombres aléatoires physiques (TRNG) extraient le caractère aléatoire de processus physiques qui se comportent d'une manière fonda-

mentalement non déterministe, ce qui en fait d'eux, les meilleurs candidats pour la génération de véritables nombres aléatoires. Il s'agit d'un matériel physique séparée de l'ordinateur et généralement connecté à celui-ci via un bus USB ou PCI. Les exemples de processus physiques utilisés pour générer les valeurs aléatoires : le bruit de Johnson [60], le bruit de Zener [61] . On peut classer les TRNG en 4 familles à savoir :

* Générateur aléatoires à base des bruits
* Générateur à oscillations libres
* Générateurs à base du chaos
* Générateurs aléatoires quantiques

L'arbre généalogique peut être vue comme ci-dessous [62].

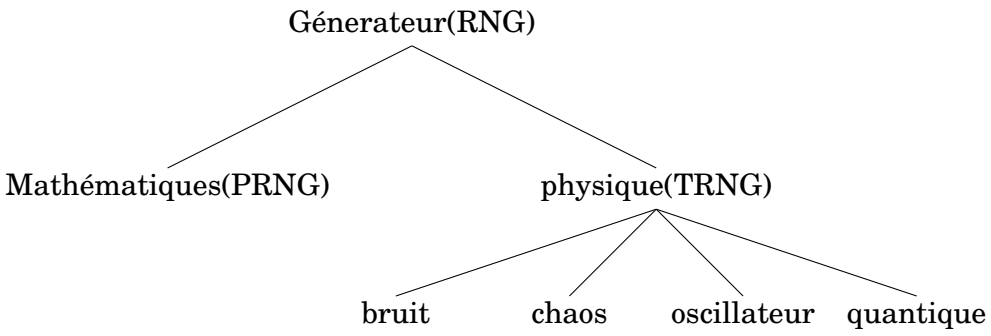


FIGURE 1.14 – Classification des générateurs de nombres aléatoires

Nous pouvons dire, qu'un vrai RNG ne doit pas être confondue à un générateur de nombres pseudo-aléatoires implémenté sur hardware ; car un tel générateur est toujours un PRNG puisqu'il s'agit tout simplement d'une implémentation matérielle.

1.3.3) Cryptographie

Depuis,quelques années, le domaine de la cryptographie a été employé pour offrir les besoins de sécurité tel que : la confidentialité, authentification, l'intégrité du message, et la non répudiation [63].

Le terme cryptographie a été inventé en 1958 par le Physicien Britannique Thomas Browne. Il provient de deux mots grecs anciens « Kruptos » qui signifie « cacher » et « graphein » qui signifie « écrire » . Ce qui signifie littéralement, cacher l'écriture. La cryptographie peut donc se définir, comme une technique permettant d'envoyer des données de manière confidentielle

sur un support. Ces données échangées peuvent être sous différents formats tels que : textes, images, vidéos, audio, etc.

La cryptanalyse quant à elle est l'ensemble de techniques utilisées pour analyser et rompre une communication sécurisée.

Le processus permettant de transformer un message en clair (message à crypter) en un message non compréhensible pour une tiers personne est appelée **cryptage** et le processus inverse est **le décryptage**.

D'après Menezes et al, la cryptographie est divisé en deux domaines d'étude : la cryptographie à clef symétrique et la cryptographie a clef asymétrique. La cryptographie à clef symétrique comme son nom l'indique, utilise une clef identique pour effectuer le opérations de chiffrement et de déchiffrement. D'autre part, la cryptographie à clef asymétrique, utilise deux clefs : tel que, une clef appelée clef publique est utilisée pour chiffrer le message , la seconde clef appelée clef privée quant à elle, est utilisée pour déchiffrer le message. Dans les sections suivantes, nous allons présenter en bref,ces 2 domaines.

1.3.3.1) *Cryptographie à clef publique : chiffrement asymétrique*

Le principe peut être illustré sur le schéma ci-contre

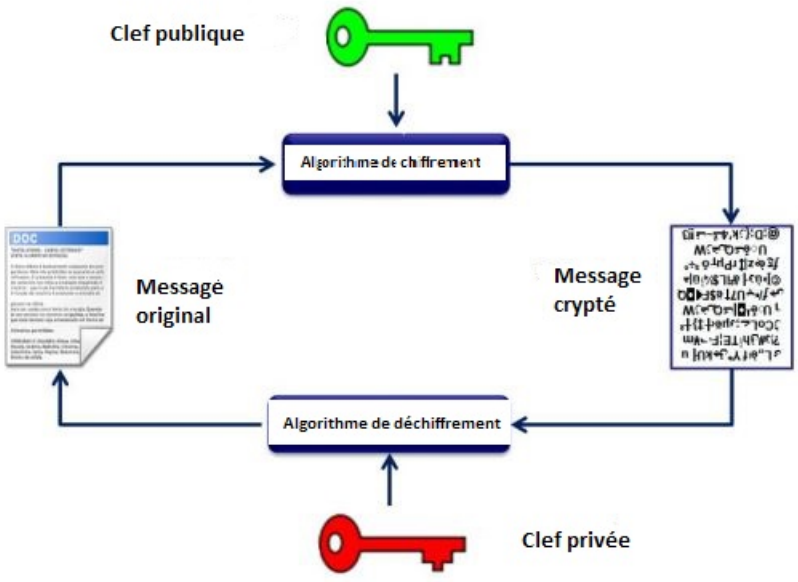


FIGURE 1.15 – Cryptographie à clef publique

Comme nous pouvons le remarquer sur la figure, le principe de chiffrement s'effectue en utilisant une paire de clef : une clef pour le processus de chiffrement (clef en vert) appelée clef publique et une autre pour le processus inverse (clef en rouge) appelée clef privée. Le principe d'échange est le suivant : un expéditeur X envoie le message chiffré au destinataire Y en utilisant une clef publique et un algorithme de chiffrement. Parallèlement, le destinataire Y doit déchiffrer ce message en utilisant sa clef à lui (clef privée) et un algorithme de déchiffrement [64, 65]. Les équations traduisant les processus de chiffrement et de déchiffrement sont données par :

$$E_{PK}(M) = C \tag{1.20}$$

$$D_{SK}(C) = M \tag{1.21}$$

où,

- PK représente la clef publique (public key)
- SK, la clef sécrète (secret key)
- E c'est la fonction qui effectue le chiffrement
- D celle qui effectue le déchiffrement
- M, message d'origine
- C, le message chiffré

Parmi les algorithmes asymétriques, nous pouvons citer : l'algorithme de RSA , la clef publique de Elgamal, la clef publique de Rabin [66, 67, 68].

1.3.3.2) *Cryptographie à clef symétrique : chiffrement symétrique*

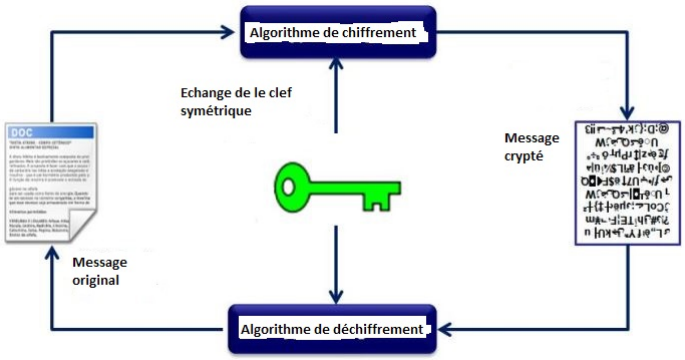


FIGURE 1.16 – Cryptographie à clef symétrique

Le chiffrement à clef symétrique utilise une seule clef pour effectuer les opérations de chiffrement et de déchiffrement de l'information [64]. Dans ce cas, l'émetteur X envoie une information chiffrée au destinataire Y à partir d'un algorithme de chiffrement et en utilisant une clef K_1. Ainsi, le destinataire Y pourra déchiffrer le message provenant de X à partir d'un algorithme de déchiffrement et en utilisant la clef identique à X [66, 67]. Les interlocuteurs pourraient donc échanger la conversation de part et d'autre en utilisant un même algorithme de chiffrement et de déchiffrement, en utilisant les clefs identiques. Parmi les algorithmes symétriques connus, nous pouvons citer : DES, IDEA, AES, Blowfish [69, 70, 71, 72]. Les équations relatives à ces opérations sont les suivantes :

$$E_{K_1}(M) = C \tag{1.22}$$

$$D_{K_1}(M) = M \tag{1.23}$$

où K_1 représente la clef identique.
Le chiffrement à clef symétrique peut être encore subdivisé en chiffrement par bloc et en chiffrement par flux.

Le chiffrement par bloc est une méthode de chiffrement qui divise le texte en clair en blocs de même taille puis les chiffrer de façon sérielle (c'est à dire l'un après l'autre) en utilisant une clef unique [64, 73]. L'opération de décryptage, consiste à diviser le texte chiffré en blocs de même taille, puis déchiffrer ces blocs l'un après l'autre afin d'obtenir le texte en clair (message déchiffré) en utilisant la clef identique que celle du chiffrement [74]. Un chiffrement par bloc est considéré comme totalement attaqué où intercepté, si la clef sécrète est récupérée par un hacker où un intrus. Dans le cas où une partie du texte est récupéré, on dira qu'il est partiellement attaqué [75]. Dans un algorithme de chiffrement bien conçu, le texte en clair et le texte chiffré doivent être statistiquement indépendants pour offrir une sécurité élevée. Dans plusieurs algorithmes de chiffrement, le chiffrement par blocs est considéré comme l'une des meilleurs techniques de chiffrement les plus utilisés. Parmi les algorithmes de chiffrement par blocs connus, nous pouvons citer : DES, 3-DES, IDEA, FEAL, Blowfish et AES. Pour évaluer un chiffrement par blocs, plusieurs critères doivent être prises en considération comme :
- La taille de la clef : une clef de grande taille est plus sécurisée que celle de petite taille bien qu'elle occupe un peu d'espace de stockage,et allonge un peu le temps de calcul.
- La taille des blocs : le choix de la taille des blocs influence sur la sécurité de l'algorithme. En utilisant les blocs de grande taille, la sécurité est plus élevée mais l'implémentation devient

un peu plus couteux.

- La complexité de l'algorithme : lorsque l'algorithme est plus complexe, les performances de celui-ci diminue. Un algorithme de chiffrement doit être le moins complexe possible pour faciliter son utilisation.

- Le niveau de sécurité : plusieurs tests statistiques permettent d'évaluer le niveau de sécurité de l'algorithme.

Toutefois, il existe aussi plusieurs techniques pour effectuer les chiffrements en bloc. Ces techniques peuvent être représentées sur les figures ci-contre :

1) Electronic Codebook (ECB)

C'est un mode de chiffrement basique, qui consiste à chiffrer chaque bloc séparément, sans chevauchement entre les blocs [6]. Le principal avantage de ce mode est que la fonction de cryptage et de décryptage peut être mise en œuvre en mode séquentiel uniquement.

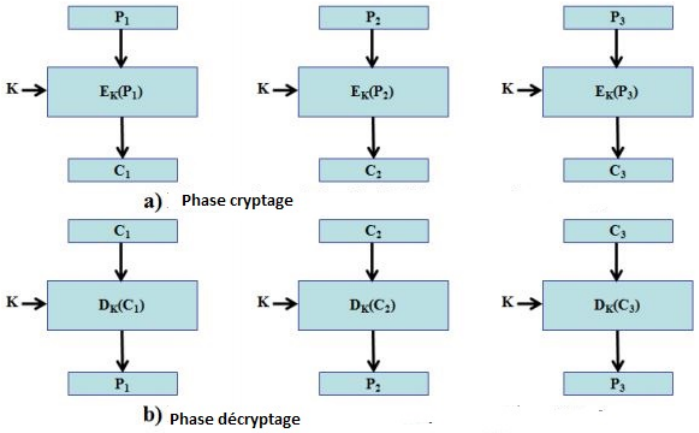


FIGURE 1.17 – Mode ECB : a) phase cryptage ; b) phase décryptage [3, 4, 5]

2) Mode Cipher-block Chaining (CBC)

En mode CBC, chaque bloc du texte en clair, effectue une opération XOR avec un bloc précédent déjà chiffré [6]. Ainsi, ce qui implique que, chaque bloc chiffré dépendra des blocs antérieurs et à chaque unique message correspondra son message chiffré. Ce mode de fonctionnement est le plus utilisé. L'inconvénient principal de ce mode est que la fonction de cryptage ne peut être mise en œuvre qu'en mode séquentiel,tandis que la fonction de décryptage est implémentée en parallèle. La figure qui illustre ce mode de chiffrement est donnée ci-dessous (voir fig 1.16).

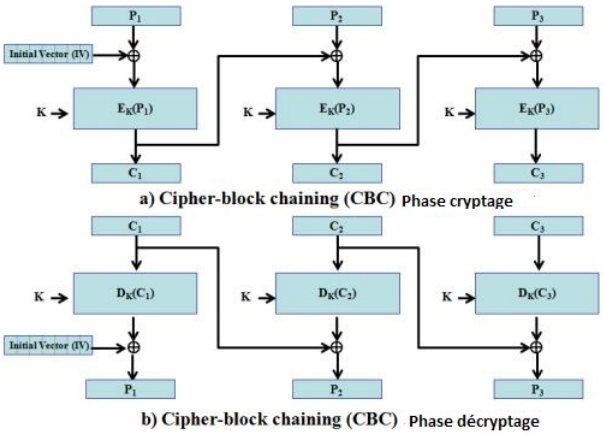


FIGURE 1.18 – Mode CBC : a) Phase de cryptage ; b) phase de décryptage [6]

3) Mode Cipher Feedback (CFB)

Dans ce mode, le vecteur initial et la clef sécrète sont traités par la fonction de cryptage pour donner un message chiffré . Ensuite on effectue le XOR du message chiffré avec un bloc du message en clair pour obtenir un autre message chiffré et ainsi de suite jusqu'à épuisement des blocs [76].

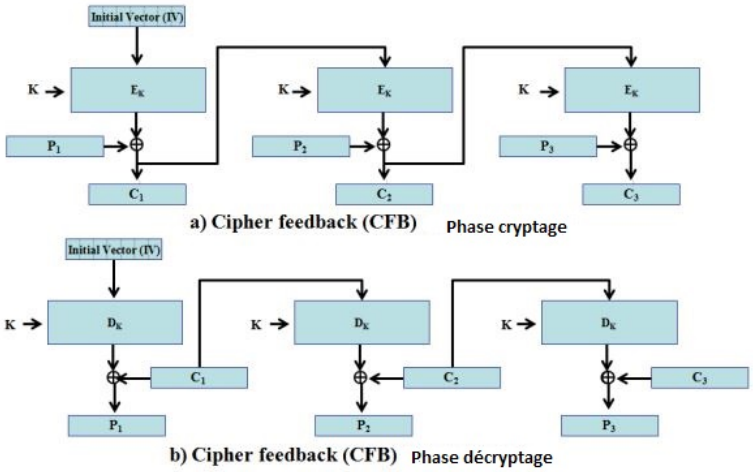


FIGURE 1.19 – Mode CFB : a) Phase de cryptage ; b) phase de décryptage [7]

4) Mode Output Feedback (OFB)

Le principe de fonctionnement est similaire à celui du mode CFB présenté précédemment dans le cas du chiffrement [6]. La différence ici s'observe au niveau de la phase de déchiffrement, car le flux entrant vers les étapes ultérieures est indépendant du message en clair. Le principe de fonctionnement est donné comme illustré à la figure (1.18 a).

5) Mode Propagating Cipher-block Chaining (PCBC)

En mode PCBC, le premier bloc de texte en clair est soumis à une opération XOR avec le vecteur d'initialisation et ensuite traité par la fonction de chiffrement pour obtenir le bloc de texte chiffré correspondant.Dans le deuxième bloc, les blocs de texte en clair et de texte chiffré sont soumis à une opération XOR et le résultat est ensuite soumis à une opération XOR avec le texte en clair du bloc actuel [77] . Le principe de fonctionnement est donné à la figure (1.18 b). Dans ce mode de fonctionnement, la modification d'un ou de plusieurs bits dans les blocs de texte en clair affectera les résultats du texte chiffré de ce bloc et de tous les blocs suivants. De plus, le changement d'un bit dans le texte chiffré affecte le bloc actuel et le bloc suivant.Ce mode a été conçu pour propager à l'infini l'effet de la modification d'un ou plusieurs bits dans le processus de cryptage/décryptage. Par conséquent, le cryptage et le décryptage de ce mode ne peuvent être mis en œuvre qu'en mode séquentiel. Toutefois, le mode de chiffrement PCBC présente les propriétés suivantes [78, 79] :

* Le texte en clair identique à une paire de clef et le vecteur initial produit un texte chiffré identique.

* Chaque bloc est chiffré en fonction des résultats des blocs chiffrés précédents. Par conséquent, la réorganisation des positions des blocs de chiffrement affectera le résultat du décryptage.

* Une erreur dans un bloc crypté affecte les résultats cryptés suivants et entraine la dégradation de l'ensemble du texte en clair.

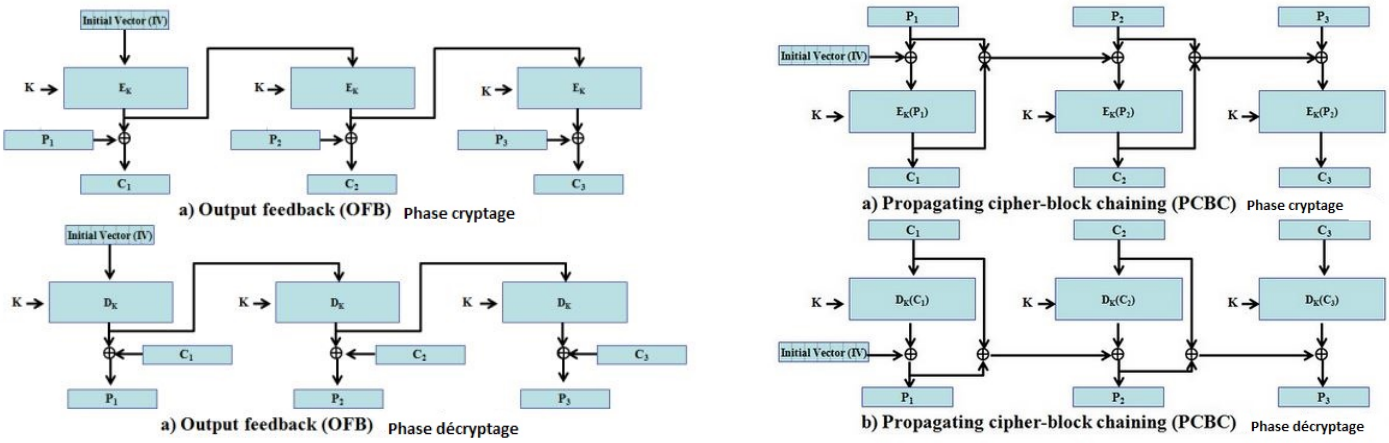


FIGURE 1.20 – Mode de fonctionnement OFB et PCBC

Le chiffrement par flux est l'une des méthodes de cryptage basées sur la clé secrète qui crypte un flux de données, comme un bit où un octet à la fois [80]. Il chiffre des caractères individuellement, tandis que le chiffrement par blocs chiffre des groupes de caractères simultanément. Habituellement, le chiffrement par blocs est utilisé pour chiffrer de grands blocs (par exemple,$n \geq 64$), tandis que le chiffrement par flux est utilisé pour chiffrer de petites unités [81]. En général, le chiffrement par flux génère des séquences de bits pseudo-aléatoires qui peuvent être soumises à un XOR avec le texte en clair pour produire le texte chiffré correspondant ; le chiffrement par flux est beaucoup plus simple que le chiffrement par blocs et plus rapide à mettre en œuvre sur le plan matériel. Un chiffrement par blocs est une méthode de cryptage basée sur la clé secrète qui crypte un bloc (par exemple 64 bits) de données en même temps ; un chiffrement par blocs traite des blocs de texte en clair de n bits en entrée (groupe de bits) pour produire n bits de texte chiffré en sortie. Ainsi, le processus de chiffrement par bloc doit être réversible afin de déchiffrer les données cryptées.

1.3.4) Cryptographie des images

Les séquences chaotiques sont des séquences pseudo-aléatoires ayant de bonnes performances et peuvent être obtenues à partir d'une équation dynamique où un système d'équations linéaires où non. En raison de bonnes caractéristiques de celles ci, elles ont été largement utilisé dans le domaine de la cryptographie des images. Au cours des dix dernières années, plusieurs variétés de systèmes chaotiques [82, 83] ont été utilisés pour concevoir les cryptosystèmes d'images tels que la récurrence logistique 2D [84], la recurrence standard 2-D [85], la recurrence d'Henon 2-D [86], la récurrence 3-D de baker [87], la récurrence 3-D LCA [88],

la récurrence 3-D du chat d'Arnold [89]. Plusieurs algorithmes de chiffrement des images basés sur les systèmes où les récurrences chaotiques sont généralement constitués de 2 étapes principales à savoir : la permutation et la diffusion. Ces 2 opérations de l'image sont souvent répétées plusieurs fois dans le cryptosystème pour accroitre le niveau de sécurité de celui-ci [90].

La permutation où la confusion est une opération dans le chiffrement des images qui consiste à modifier les différentes positions des pixels contenues dans celle ci. La diffusion quant à elle consiste à modifier les valeurs de pixels contenues dans celle-ci. La permutation peut également se scinder en 2 catégories : permutation de pixels et permutation de bits .

Dans [91], une nouvelle méthode de permutation de matrice de bits en 3-D est proposée, en utilisant le système de Chen qui produit des valeurs aléatoires pour permuter les bits de l'image de départ (image que l'on veut crypter). En combinant le système de Chen avec la récurrence 3-D du chat, dans le processus de permutation, une nouvelle règle de permutation intervient. Dans ce processus, nous avons une permutation qui s'effectue au niveau des pixels contrairement à ce qui s'effectue lorsqu'on fait sur les bits comme dans le cas de Chen ; la permutation au niveau des bits prend 8 fois plus de temps que celle au niveau des pixels. Ceci est due au fait que les pixels d'une image est codée sur 8 bits.

Cao et al [92] présente une méthode de diffusion de pixels en utilisant une matrice en 2-D qu'on transforme ensuite en un vecteur pour avoir des valeurs de pixels sur une dimension. Malheureusement, cette technique de chiffrement n'est pas optimale, car elle consomme en temps de calcul, de plus elle nécessite un espace de stockage pour ces valeurs diffusées.

Dans cet optique, Ye et Huang [93] propose donc une méthode de diffusion par blocs qui réduit le temps de calcul, mais la technique de chiffrement reste faible face à certaines attaques informatiques.

Afin d'améliorer la robustesse d'un cryptosystème face à certains attaques informatiques, Yu et al [94] propose un algorithme de cryptage d'images basé sur la transformée de fourrier fractionnelle à temps court tronquée en phase (en anglais phase-truncated short time fractional Fourier transform "PTSTfrFT") et un système hyper-chaotique. Un système de rétro-action est utilisé pour concevoir l'opération de diffusion qui contribue à l'amélioration des capacités de non- interférence du système.

Le cryptage joue un rôle fondamental dans la sécurisation des biens informatiques et repose sur 4 éléments fondamentaux suivant : la confidentialité (encode le contenu du message), l'authentification (vérifie l'origine du message), l'intégrité (prouve que l'intégrité du message n'a

pas été modifié depuis son envoi), la non-répudiation (empêche l'expéditeur de nier d'avoir envoyé un message crypté).

La cryptanalyse en outre, est la science qui étudie les méthodes de déchiffrement du texte chiffré sans connaître la clé utilisée ; cela dépend de la nature de l'algorithme et d'une certaine connaissance des propriétés du texte en clair où du texte chiffré. La cryptanalyse exploite les propriétés de l'algorithme pour déduire le texte en clair ou la clé utilisée spécifique. Une fois que l'adversaire a réussi à découvrir l'identité de la clé utilisée, toutes les communications passées et futures sont vulnérables. Un algorithme d'un message codé par authentification (MAC) nécessitera un effort de cryptanalyse supérieure ou égal à l'effort de force brute. Il serait impraticable d'essayer de trouver une collision avec un espace de clés très large.

Conclusion

Dans ce chapitre, il était question pour nous de présenter premièrement, quelques récurrences où systèmes dynamiques chaotiques existants dans la littérature, puis de préciser les outils d'analyse permettant d'étudier analytiquement ceux -ci à savoir les outils qualitatifs d'une part, et les outils quantitatifs d'autre part. Il convient donc de noter que, ces systèmes où les récurrences pré-citées, présentaient quelques applications dans le domaine de la cryptographie, l'implémentation des générateurs de nombres pseudo-aléatoires etc. Enfin nous avons présenté les contributions des travaux apportés dans la cryptographie des images . En dépit de cela, il nous est donc convenu de constater, que ces récurrences présentées dans la littérature présentaient des faiblesses de sécurité, demandaient beaucoup de ressources matériels pour sa réalisation, des algorithmes cryptographiques lents en calcul etc. Ces limites présentées ci-dessous, nous conduit à apporter une contribution pour l'amélioration de la récurrence d'Arnold et son implémentation dans le domaine de la cryptographie.

CHAPITRE 2

MATÉRIELS ET MÉTHODES

Introduction

Dans le chapitre 1, nous avons présenté sommairement quelques systèmes dynamiques chaotiques et récurrences dynamiques les plus utilisées dans les applications tels que, la cryptographie, les générateurs de nombres aléatoires, en sténographie, le chaos, les jeux de hasard, etc. Compte tenu, des failles où des faiblesses relevées sur celles ci, il convient donc d'apporter une nouvelle approche sur la récurrence d'Arnold en ajoutant des perturbations sur celle-ci afin d'accroitre significativement sa période. Pour mener à bien cet étude, nous allons dans ce chapitre, présenter une étude détaillée de la récurrence d'Arnold en dimension 4 sur 4 bits, puis sur 2 bits en insérant les variables de perturbations.

2.1) Présentation de la récurrence d'Arnold discrète

La recurrence d'Arnold [95], est un système chaotique discret très répandu qui brouille les pixels d'une image quelconque prise en entrée pour obtenir en sortie la même image après un certain nombre d'itérations noté P. Le paramètre P est définit ici comme la périodicité de la carte de transformation. La récurrence d'Arnold présente quelques propriétés tels que, le brouillage des pixels, la forte sensibilité aux conditions initiales et paramètres de contrôles.
La recurrence d'Arnold en 2-D peut etre exprimée comme suit :

$$\begin{pmatrix} x(t+1) \\ y(t+1) \end{pmatrix} = \begin{pmatrix} 1 & \alpha \\ \beta & 1+\alpha.\beta \end{pmatrix} \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} \bmod N \tag{2.1}$$

Les constantes α et β sont définit positives et représentent les paramètres de contrôles du système ($\alpha, \beta > 0$) ; les variables, $x(t), y(t) \in \{0, ..., N-1\}$; N représente la valeur de la dimension d'une image de taille $N \times N$. Cette récurrence est périodique et la période dépend de la valeur du modulo N et de la parité des paramètres de contrôles α et β ; lorsque $\alpha = \beta = 1$ et $N = 2^p$, le calcul des périodes est donné par l'équation suivante :

$$\Pi_p = 2.\Pi_{p-1}, p > 2 \tag{2.2}$$

Avec $\Pi_1 = \Pi_2 = 3$, étant la période minimale, et p représente le nombre de bits (la précision de calcul) [96].

2.2) Récurrence d'Arnold linéaire par morceaux en dimension 4 (RALM 4D)

La récurrence discrète d'Arnold 2-D présentée ci-dessus, peut être étendue en dimension 4, en conservant les mêmes propriétés et caractéristiques que celle de la récurrence discrète 2-D. Ces propriétés sont : la sensibilité aux conditions initiales et paramètres de contrôles, le déterminisme, la conservation de l'aire ou de la surface, la réversibilité.

Pour concevoir un algorithme de chiffrement, deux opérations mathématiques sont nécessaires à savoir la confusion et la diffusion. La confusion ou la permutation, est une étape dans la procédure de chiffrement qui consiste à modifier les positions des éléments contenus dans un vecteur ou une matrice de données. Ces données sont généralement des nombres entiers naturels. L'opération de diffusion quant à elle consiste à modifier ces valeurs entières. Ces 2 opérations de chiffrement, sont en général liés respectivement par des équations mathématiques.

Ceci peut se justifier à travers le travail de Ye et Zhao [97], qui ont proposé un schéma de chiffrement d'images basé sur les récurrences affines modulaires, qui sont des extensions de générateurs congruents linéaires. Le processus de permutation utilise deux récurrences modulaires affines pour effectuer la modification des positions de pixels de l'image tandis que le processus de diffusion utilise deux autres récurrences modulaires affines pour la modification des valeurs de pixels.

Plus récemment, Ratna et al [98], proposent également un algorithme de chiffrement d'images en utilisant deux systèmes dynamiques différents. Pour effectuer la phase de permutation , la récurrence d'Arnold est utilisée tandis que la récurrence d'Henon a été utilisée pour la phase de diffusion.

Ainsi, pour améliorer les performances et les différentes techniques de chiffrement proposées dans la littérature, on propose dans ce travail de thèse, une technique particulière de chiffrement d'images. La technique consiste à effectuer simultanément les opérations de permutation et de diffusion en une seule étape en utilisant un système dynamique.

Pour effectuer ces opérations de chiffrement simultanément, il est donc nécessaire de penser à un système en dimension 4 .

Le système en dimension 4 que nous proposons est l'extension en 4-D de la récurrence d'Arnold discrète présenté précédemment. L'extension 4-D peut donc s'écrire comme :

$$\begin{cases} x(t+1) = x(t) + \alpha y(t) \\ y(t+1) = y(t) + \beta x(t+1) \\ q(t+1) = q(t) + \gamma y(t+1) \\ r(t+1) = r(t) + \zeta q(t+1) \end{cases} \text{mod } N \tag{2.3}$$

x, y q et r sont les variables d'états et les constantes α , β, γ et ζ, les paramètres de contrôles du système qui sont des entiers naturels strictement positifs (α , β, $\gamma \in \mathbb{N}^*$).

Après analyse sur le calcul des périodes de la récurrence 4-D ci-dessous, il convient, de remarquer que les périodes restent toujours faibles soit 2 fois plus grande que la période de la récurrence d'Arnold discrète.

Ainsi, dans le but de concevoir un algorithme de chiffrement , il est nécessaire d'avoir des séquences pseudo-aléatoires. Pour le faire, on apporte des pertubations non linéaires sur chaque variable d'état de la récurrence 4-D. Le système obtenu après modification est la récurrence linéaire par morceaux d'Arnold 4-D (4D-PWLCM) donné par :

$$\begin{cases} x(t+1) = x(t) + \alpha y(t) + F_1(y,t) \\ y(t+1) = y(t) + \beta x(t+1) + F_2(x,t) \\ q(t+1) = q(t) + \gamma y(t+1) + F_3(y,t) \\ r(t+1) = r(t) + \zeta q(t+1) + F_4(q,t) \end{cases} \text{mod } N \tag{2.4}$$

Les fonctions $F_1(y,n)$, $F_2(x,n)$, $F_3(y,n)$ et $F_4(z,n)$ sont les termes de pertubations . Les expressions mathématiques de ces termes sont données par :

$$\begin{cases} F_1(y,t) = \sum_{i=1}^{M} (a_i + y(t)) \bmod c_i \\ F_2(x,t) = \sum_{j=1}^{N} (b_j + x(t+1)) \bmod d_j \\ F_3(y,t) = \sum_{k=1}^{P} (e_k + y(t+1)) \bmod g_k \\ F_4(q,t) = \sum_{l=1}^{W} (f_l + q(t+1)) \bmod h_l \end{cases} \tag{2.5}$$

La récurrence linéaire par morceaux d'Arnold 4-D, présentée ci-dessous est inversible comme celle de la récurrence d'Arnold conventionnelle et le système inverse est donnée par :

$$\begin{cases} r(t) = r(t+1) - \zeta q(t+1) - F_4(q,t) \\ q(t) = q(t+1) - \gamma y(t+1) - F_3(y,t) \\ y(t) = y(t+1) - \beta x(t+1) - F_2(x,t) \\ x(t) = x(t+1) - \alpha y(t) - F_1(y,t) \end{cases} \bmod N \tag{2.6}$$

2.3) Stabilité de la RALM 4D

2.3.1) Etude sur 4 bits

2.3.1.1) *Calcul de la matrice Jacobienne*

La récurrence d'Arnold linéaire par morceaux en dimension 4 (RALM 4D) sur 4 bits est donnée par le système d'équation suivant :

$$\begin{cases} x(t+1) = x(t) + \alpha y(t) + F_1(y,t) \\ y(t+1) = y(t) + \beta x(t+1) + F_2(x,t) \\ q(t+1) = q(t) + \gamma y(t+1) + F_3(y,t) \\ r(t+1) = r(t) + \zeta z(t+1) + F_4(q,t) \end{cases} \bmod 2^4 \tag{2.7}$$

$F_1(y,t)$, $F_2(x,t)$, $F_3(y,t)$ et $F_4(q,t)$ sont les fonctions de pertubations comme définit à (2.5).
La matrice jacobienne est une matrice contenant les dérivées partielles du premier ordre d'une fonction vectorielle en un point donné. C'est un outil le plus souvent employé pour effectuer l'étude analytique des systèmes dynamiques .
Les éléments de la matrice jacobienne de la RALM 4D proposée ci-dessous, sont donnés par :

$$J = \begin{pmatrix} \frac{\partial x(t+1)}{\partial x} & \frac{\partial x(t+1)}{\partial y} & \frac{\partial x(t+1)}{\partial q} & \frac{\partial x(n+1)}{\partial r} \\ \frac{\partial y(t+1)}{\partial x} & \frac{\partial y(t+1)}{\partial y} & \frac{\partial y(t+1)}{\partial q} & \frac{\partial y(t+1)}{\partial r} \\ \frac{\partial q(t+1)}{\partial x} & \frac{\partial q(t+1)}{\partial y} & \frac{\partial q(t+1)}{\partial q} & \frac{\partial q(t+1)}{\partial r} \\ \frac{\partial r(t+1)}{\partial x} & \frac{\partial r(t+1)}{\partial y} & \frac{\partial r(t+1)}{\partial q} & \frac{\partial r(t+1)}{\partial r} \end{pmatrix} \tag{2.8}$$

En introduisant la fonction de Heaviside comme présenté dans [59], le système d'équation donné à (2.5) peut encore s'écrire comme :

$$\begin{cases} F_1(y,t) = \sum_{i=1}^{M} (a_i + y(t) - q_i c_i u(a_i + y(t) - c_i)) \\ F_2(x,t) = \sum_{j=1}^{N} (b_j + x(t+1)) - q_j d_j u(b_j + x(t+1) - d_j)) \\ F_3(y,t) = \sum_{k=1}^{P} (e_k + y(t+1) - q_k g_k u(c_k + y(t+1) - g_k)) \\ F_4(q,t) = \sum_{l=1}^{W} (f_l + z(t+1) - q_l h_l u(f_l + z(t+1) - h_l)) \end{cases} \tag{2.9}$$

En dérivant les expressions des éléments de la matrice jacobienne données à (2.8), on obtient :

$$J = \begin{pmatrix} 1 & J_{12}(t) & 0 & 0 \\ J_{21}(t) & J_{22}(t) & 0 & 0 \\ J_{31}(t) & J_{32}(t) & 1 & 0 \\ J_{41}(t) & J_{42}(t) & J_{43}(t) & 1 \end{pmatrix} \tag{2.10}$$

On a :

$J_{12}(t) = \frac{\partial x(t+1)}{\partial y} = \alpha + M - \sum_{i=1}^{M} q_i c_i \delta(a_i + y(t) - c_i)$; or $\delta(a_i + y(t) - c_i) = \begin{cases} 1, \text{ si } q_i = 1 \\ 0 \text{ ailleurs .} \end{cases}$;

Où,

$$q_i = \lfloor \frac{a_i + y(t)}{c_i} \rfloor \tag{2.11}$$

Ceci nous permet donc d'écrire :

$$J_{12}(t) = \alpha' - \sum_{i=1}^{M} c_i \delta(y(t) + a_i - c_i) \tag{2.12}$$

où, $\alpha' = \alpha + M$;

$$J_{21}(t) = \frac{\partial y(t+1)}{\partial x} = \beta' - \sum_{j=1}^{N} d_j \delta(x(t+1) + b_j - d_j) \tag{2.13}$$

Avec $\beta' = \beta + P$

$J_{22}(t) = \frac{\partial y(t+1)}{\partial y} = 1 + \beta'.\frac{\partial x(t+1)}{\partial y} - \frac{\partial x(t+1)}{\partial y} \sum_{j=1}^{N} d_j \delta(x(t+1) + b_j - d_j).$

or, $J_{12}(t) = \frac{\partial x(t+1)}{\partial y}$, ainsi l'expression de $J_{22}(t)$ nous donne :

$$J_{22}(t) = 1 + J_{12}(t).(\beta' - \sum_{j=1}^{N} d_j \delta(x(t+1) + b_j - d_j) = 1 + J_{12}(t).J_{21}(t) \tag{2.14}$$

$J_{31}(t) = \frac{\partial q(t+1)}{\partial x}$

$= \gamma'.\frac{\partial y(t+1)}{\partial x} - \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k).\frac{\partial y(t+1)}{\partial x}$

$= \gamma' J_{21}(t) - J_{21}(t) \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k)$

$= J_{21}(t)(\gamma' - \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k)$

D'où,

$$J_{31}(t) = J_{21}(t)(\gamma' - \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k) \tag{2.15}$$

Avec , $\gamma' = \gamma + P$

$J_{32}(t) = \frac{\partial q(t+1)}{\partial y}$

$= \gamma'.\frac{\partial y(t+1)}{\partial y} - \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k).\frac{\partial q(t+1)}{\partial y}$

$= \gamma' J_{22}(t) - J_{22}(t). \sum_{k=1}^{P} g_k \delta(y(n+1) + e_k - g_k)$

$= J_{22}(t)(\gamma' - \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k)$

Ainsi,

$$J_{32}(t) = J_{22}(t)(\gamma' - \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k) \tag{2.16}$$

$J_{43}(t) = \frac{\partial r(t+1)}{\partial q}$

$= \zeta'.\frac{\partial q(t+1)}{\partial q} - \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l).\frac{\partial q(t+1)}{\partial q}$

$= \zeta' - \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l)$ car , $\frac{\partial q(n+1)}{\partial q} = 1$

Donc ,

$$J_{43}(t) = \zeta' - \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l) \tag{2.17}$$

avec, $\zeta' = \zeta + W$

$J_{41}(t) = \frac{\partial r(t+1)}{\partial x}$

$= \zeta' . \frac{\partial q(t+1)}{\partial x} - \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l) . \frac{\partial q(t+1)}{\partial x}$

$= \zeta' . J_{31}(t) - J_{31}(t) . \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l)$

$= J_{31}(t)(\zeta' - \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l)) = J_{31}(t) J_{43}(t)$

Donc , on a :

$$J_{41}(t) = J_{31}(t) J_{43}(t) \tag{2.18}$$

$J_{42}(t) = \frac{\partial r(t+1)}{\partial y}$

$= \zeta' . \frac{\partial q(t+1)}{\partial y} - \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l) . \frac{\partial q(t+1)}{\partial y}$

$= \zeta' . J_{32} - J_{32} . \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l)$

$= J_{32}(t) . J_{43}(t)$

Ainsi,

$$J_{42}(t) = J_{32}(t) . J_{43}(t) \tag{2.19}$$

Nous pouvons donc ainsi récapituler tous les éléments de la matrice jacobienne à travers le système suivant :

$$\begin{cases} J_{11} = 1 \\ J_{12}(t) = \alpha' - \sum_{i=1}^{M} c_i \delta(y(t) + a_i - c_i) \\ J_{13} = J_{14} = 0 \\ J_{21}(t) = \beta' - \sum_{j=1}^{N} d_j \delta(x(t+1) + b_j - d_j) \\ J_{22}(t) = 1 + J_{12}(t) J_{21}(t) \\ J_{23} = J_{24} = 0 \\ J_{31}(t) = J_{21}(t)(\gamma' - \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k) \\ J_{32}(t) = J_{22}(t)(\gamma' - \sum_{k=1}^{P} g_k \delta(y(t+1) + e_k - g_k) \\ J_{33} = J_{44} = 1 \\ J_{43}(t) = \zeta' - \sum_{l=1}^{W} h_l \delta(q(t+1) + f_l - h_l) \\ J_{41}(t) = J_{31}(t) J_{43}(t) \\ J_{42}(t) = J_{32}(t) J_{43}(t) \end{cases} \tag{2.20}$$

Pour valider que la RALM 4-D est conservatif comme la récurrence d'Arnold en 2-D, calculons le déterminant de la matrice jacobienne présenté ci-dessous.

$$\text{On a : } \triangle = det(J) = \begin{vmatrix} 1 & J_{12}(t) & 0 & 0 \\ J_{21}(t) & J_{22}(t) & 0 & 0 \\ J_{31}(t) & J_{32}(t) & 1 & 0 \\ J_{41}(t) & J_{42}(t) & J_{43}(t) & 1 \end{vmatrix} = \begin{vmatrix} 1 & J_{12}(t) & 0 \\ J_{21}(t) & J_{22}(t) & 0 \\ J_{31}(t) & J_{32}(t) & 1 \end{vmatrix} = \begin{vmatrix} 1 & J_{12}(t) \\ J_{21}(t) & J_{22}(t) \end{vmatrix}$$

$= J_{22}(t) - J_{21}(t) J_{12}(t)$

$= 1 + J_{12}(t) J_{21}(t) - J_{21}(t) J_{12}(t)$

$= 1$

On obtient donc finalement,

$$\triangle = 1 \tag{2.21}$$

Ceci implique que, la récurrence d'Arnold linéaire par morceaux en dimension 4 que nous proposons est conservatif et peut être utile pour un algorithme de chiffrement d'images.

2.3.1.2) *Calcul des valeurs propres*

La RALM 4-D est conservative car , det(J)=1 ; ceci implique que la somme des exposants de Lyapunov doit être nulle.

Pour cela, déterminons d'abord les valeurs propres de la matrice jacobienne issue de la RALM 4D que nous proposons dans ce travail de thèse.

Le polynome caractéristique de la matrice jacobienne est :

$$P_J(\lambda) = det(J - \lambda I_4) = \begin{vmatrix} 1-\lambda & J_{12}(t) & 0 & 0 \\ J_{21}(t) & J_{22}(t)-\lambda & 0 & 0 \\ J_{31}(t) & J_{32}(t) & 1-\lambda & 0 \\ J_{41}(t) & J_{42}(t) & J_{43}(t) & 1-\lambda \end{vmatrix}$$

$$= \begin{vmatrix} 1-\lambda & J_{12}(t) & 0 \\ J_{21}(t) & J_{22}(t)-\lambda & 0 \\ J_{31}(t) & J_{32}(t) & 1-\lambda \end{vmatrix}$$

$$= \begin{vmatrix} 1-\lambda & J_{12}(t) \\ J_{21}(t) & J_{22}(t)-\lambda \end{vmatrix}$$

$$= (1-\lambda)^2[(1-\lambda)(J_{22}(t)-\lambda) - J_{12}(t)J_{21}(t)]$$

Ainsi,

$$P_5(\lambda) = (1-\lambda)^2[(1-\lambda)(J_{22}(t)-\lambda) - J_{12}(t)J_{21}(t)] \tag{2.22}$$

Recherchons les solutions du polynôme caractéristique $P_5(\lambda)$ pour déterminer les valeurs propres.

$P_5(\lambda) = 0 \Longrightarrow$

$$\begin{cases} (\lambda-1)^2 = 0 \ \ (1) \\ o\grave{u} \\ (1-\lambda)(J_{22}(t)-\lambda) - J_{12}(t)J_{21}(t) = 0 \ \ (2) \end{cases}$$

D'après (1), $\lambda_1 = \lambda_2 = 1$

Ainsi,

$$\lambda_1 = \lambda_2 = 1 \tag{2.23}$$

$(2) \Longrightarrow J_{22}(t) - \lambda - J_{22}(t)\lambda + \lambda^2 - J_{12}(t)J_{21}(t) = 0$

$\Longrightarrow \lambda^2 - (1 + J_{22}(t))\lambda + J_{22}(t) - J_{12}(t)J_{21}(t) = 0$

Ainsi, les solutions sont données par :

$\lambda_3 = \frac{(1+J_{22}(t))+\sqrt{(J_{22}(t)-1)^2+4J_{12}(t)J_{21}(t)}}{2}$

et $\lambda_4 = \frac{(1+J_{22}(t))-\sqrt{(J_{22}(t)-1)^2+4J_{12}(t)J_{21}(t)}}{2}$

or, $J_{22}(t) = 1 + J_{12}(t)J_{21}(t)$

Ainsi,

$$\lambda_3 = 1 + \frac{1}{2}J_{12}(t)J_{21}(t)(1 + \sqrt{1 + \frac{4}{J_{12}(t)J_{21}(t)}}) \tag{2.24}$$

et

$$\lambda_4 = 1 + \frac{1}{2}J_{12}(t)J_{21}(t)(1 - \sqrt{1 + \frac{4}{J_{12}(t)J_{21}(t)}}) \tag{2.25}$$

Avec $J_{12}(t)$ et $J_{21}(t)$, les expressions données par les équations (2.12) et (2.13).

2.3.1.3) *Calcul des exposants de Lyapunov*

La méthode de calcul des exposants de lyapunov s'effectue en utilisant la méthode d'orthogonalisation de Gram-Schmidt (GSR) [41]. Après observation des valeurs propres calculées plus haut, nous déduisons que nous aurons 2 exposants de lyapunov qui valent 0 à savoir $lyap1 = lyap2 = log(1) = 0$. Les 2 autres exposants de lyapunov sont donnés par : $lyap3 = log(\lambda_3)$ et $lyap4 = log(\lambda_4)$. Un système est dit chaotique lorsqu'il possède au moins un exposant de lyapunov qui est supérieur à 0. En itérant la récurrence d'Arnold linéaire par morceaux en dimension 4, ainsi que la matrice jacobienne donnée ci-dessous, on obtient 4 valeurs des exposants correspondant pour chaque condition initiale x_0 , y_0 $(0 < x_0, y_0 < 15)$.
En faisant varier les conditions initiales x_0 et y_0, on fixe q_0 et r_0 pour étudier l'évolution au

cours du temps de la RALM 4D sur 4 bits .

Comme nous voulons effectuer le chiffrement des images, ceci implique d'utiliser ce système comme outil de chiffrement des pixels d'une image . En effet, le pixel est l'élément de base d'une image . Nous pouvons donc l'observer sur le schéma ci-dessous.

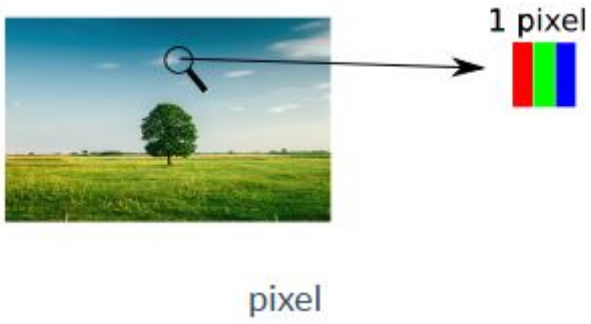


FIGURE 2.1 – section d'un pixel sur une image

a) Exposant de lyapunov en fonction des conditions initiales

Soit un pixel z_0 codée sur 8 bits et représenté en binaire par :
$z_0 = (x_7x_6x_5x_4x_3x_2x_1x_0)_2$ avec x_7 à x_0 les valeurs binaires représentant respectivement les bits de poids forts et faibles.
Dans ce cas de figure nous voulons effectuer le chiffrement en utilisant comme précision 4 bits. Ceci conduit à utiliser 2 variables sur la RALM 4D car chacune d'elle est codée sur 4 bits. Les variables que nous utilisons sont les variables x et y données par les conditions initiales x_0 et y_0. Les conditions initiales q_0 et r_0 sont fixées pour faire l'étude des exposants de lyapunov.
Ainsi, la décomposition d'un pixel sur 4 bits en fonction des conditions initiales x_0 et y_0 peut s'écrire comme suit :
$z_0 = y_0(2^4)^0 + x_0(2^4)^1$
$= x_0.16 + y_0$

Pour observer donc l'influence de ces conditions initiales sur les pixels d'une image on pose :

$$z_0 = x_0 2^n + y_0 \tag{2.26}$$

où, n représente ici le nombre de bits ; dans ce cas, n=4 bits. x_0 et y_0 sont les conditions initiales codées respectivement sur 4 bits car, l'étude dynamique est faite sur une précision de 4 bits.

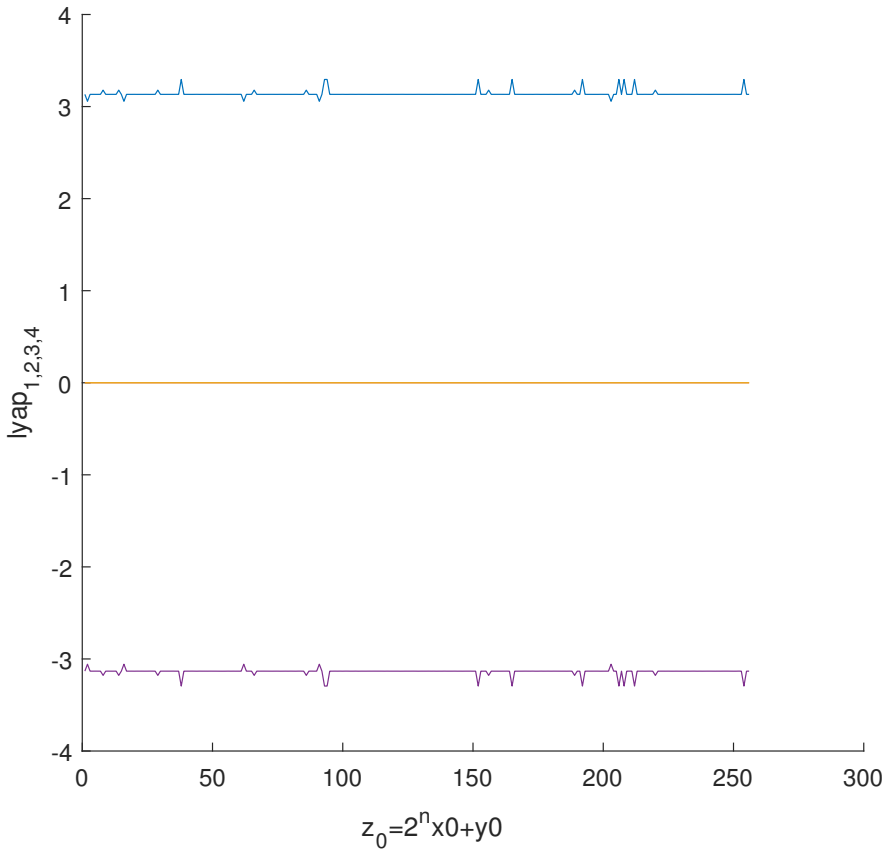


FIGURE 2.2 – Evolution des exposants de lyapunov pour différentes combinaisons de conditions initiales x_0 et y_0 en utilisant la 4D-PWLCM

Pour représenter ces exposants, nous avons fixé les paramètres de contrôles suivants :

$a = e = (1,1,0,0)$; $c = g = (0,3,1,1)$; $d = h = (3,5,1,1)$; $b = f = (0,2,0,0)$

Les conditions initiales sur q et r sont prises tel que : $q_0 = r_0 = 1$

Nous observons donc sur la figure (2.2) ci dessous, que la 4D-PWLCM est chaotique, car on

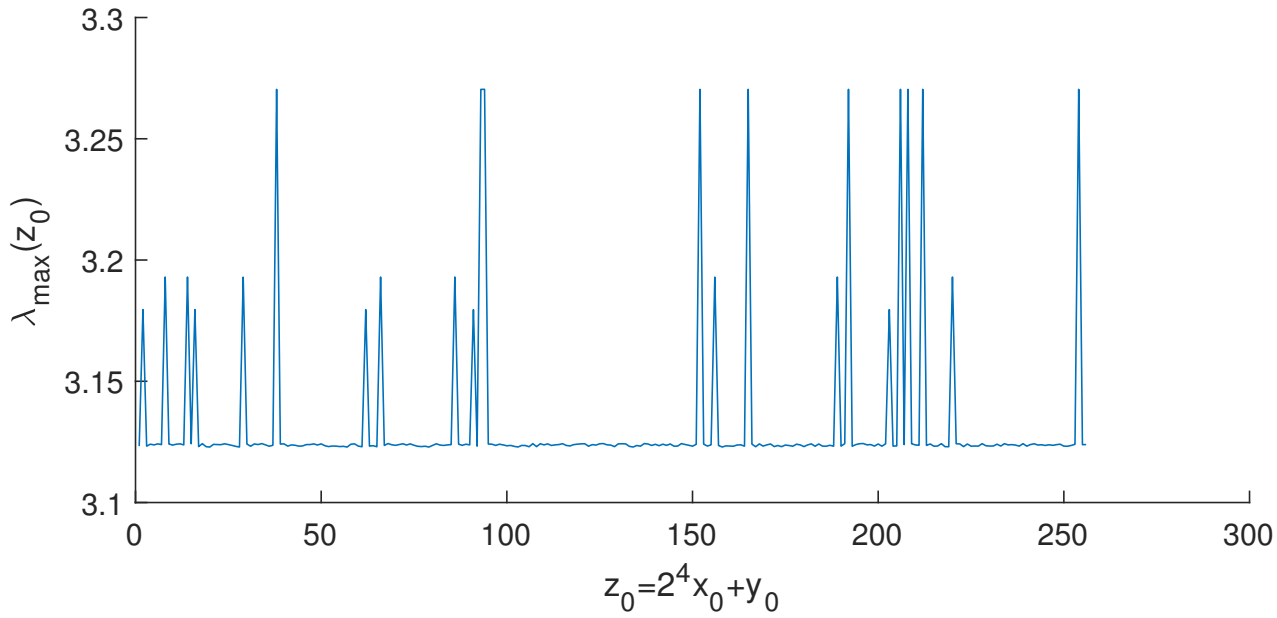


FIGURE 2.3 – Evolution du plus grand exposant de Lyapunov pour différentes combinaisons de conditions initiales x_0 et y_0 en utilisant la 4D-PWLCM

observe au moins un exposant positif. De plus cette récurrence présente un caractère conservatif comme la récurrence d'Arnold conventionnelle, car la somme de ses exposants de lyapunov valent 0. On observe également à la figure (2.3), que le plus grand exposant de lyapunov reste toujours positif quelque soit la combinaison prise pour les conditions initiales et paramètres de contrôles. Ces observations faites sur ces figures justifie donc le choix de la 4D-PWLCM sur 4 bits pour l'élaboration d'un algorithme de chiffrement.

Il est à noter aussi que cette récurrence est réversible et son système inverse a été donnée à l'équation (2.6) définit plus haut. Donc nous pouvons effectuer du chiffrement ainsi que le déchiffrement en utilisant un système en 4-D en utilisant des blocs d'images de 16×16. La taille du bloc représente ici la précision de calcul qui vaut 4 bits.

a) Exposants de lyapunov en fonction des paramètres de contrôles

Après avoir étudié le comportement des exposants de Lyapunov en fonction des conditions initiales, nous avons observé que le système garde une dynamique chaotique pour différentes valeurs de x_0, y_0, q_0 et r_0 contenues dans l'espace de phases.

Ainsi, nous voulons étudier aussi l'influence des paramètres de contrôles sur les exposants de Lyapunov . En effet, faisons varier les paramètres de contrôles a_1,a_2, a_3 dans l'espace de

phases ; ainsi, cette variation peut être définit à travers l'équation suivante :

$$a_r = \sum_{i=1}^{M} 2^{n(i-1)} a_i \tag{2.27}$$

Pour effectuer les simulations, considérons $N = 3$ (nombre de paramètres de pertubations), n=4 (nombre de bits), $\{a_i\}_{1 \leq i \leq N}$ représente les variations des coordonnées du vecteur **a** dans l'espace de phase.

La représentation des exposants de Lyapunov en fonction des variations du vecteur de contrôle **a** est donnée ci-dessous.

Nous observons également que la récurrence 4-D PWLCM est chaotique sur 4 bits ; car on a un exposant positif, un autre négatif et les 2 autres nuls. Il reste conservatif même en faisant varier les paramètres de contrôles.

Ceci convient donc une fois de plus à utiliser la récurrence 4D-PWLCM pour implémenter un algorithme de chiffrement.

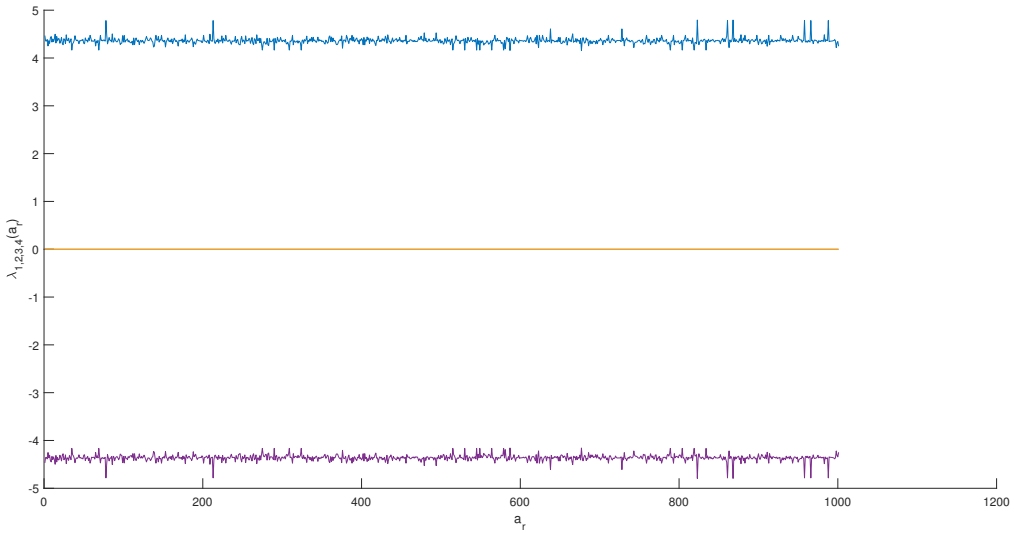


FIGURE 2.4 – Evolution des exposants de Lyapunov en variant les paramètres de contrôles **a**

2.3.2) Etude sur 2 bits

Les calculs des valeurs propres et la matrice jacobienne sur 4 bits de la 4-D PWLCM restent valables sur une précision de 2 bits ; car, le système garde la même morphologie ainsi que la même dimension. La différence intervient juste sur la précision des valeurs d'entrées ou en d'autres termes du nombre de bits.
Ainsi, pour effectuer l'étude analytique dans le cadre de 2 bits, il est judicieux d'étudier l'évolution au cours du temps des exposants de lyapunov.
Nous étudierons d'une part, les exposants de lyapunov en fonction des conditions initiales x_0,y_0,q_0 , r_0 et d'autre part, en fonction des paramètres de contrôles de la 4-D PWLCM proposé.

a) Exposant de lyapunov en fonction des conditions initiales

Nous étudions l'influence des conditions initiales x_0,y_0,q_0 et r_0 sur les exposants de lyapunov. Ces conditions initiales sont prises de 0 à 2^2-1, car elles sont représentées sur 2 bits.
Dans le cadre de 2 bits la pixel z_0 se décomposera en utilisant 4 variables codée chacune d'elle respectivement sur 2 bits.
Ainsi, en utilisant les conditions initiales , la pixel z_0 peut se décomposer comme suit :

$z_0 = x_0.(2^2)^0 + y_0.(2^2)^1 + q_0.(2^2)^2 + r_0.(2^2)^3$
$= x_0 + 2^n y_0 + q_0 2^{2n} + r_0 2^{3n}$

Ainsi, on obtient :

$$z_0 = x_0 + 2^n y_0 + q_0 2^{2n} + r_0 2^{3n} \tag{2.28}$$

où, n=2 bits ; les conditions initiales x_0, y_0,q_0 et r_0 sont prises de 0 à 3 .

Nous représentons donc l'évolution au cours du temps des exposants de Lyapunov en fonction de z_0 .

Pour représenter ces exposants, les paramètres de contrôles sont fixés comme :

$a = e = (1,1,0,0)$; $c = g = (0,3,1,1)$; $d = h = (3,3,1,1)$; $b = f = (0,2,0,0)$
Nous pouvons également remarquer que, sur 2 bits, la 4D-PWLCM présente un caractère chao-

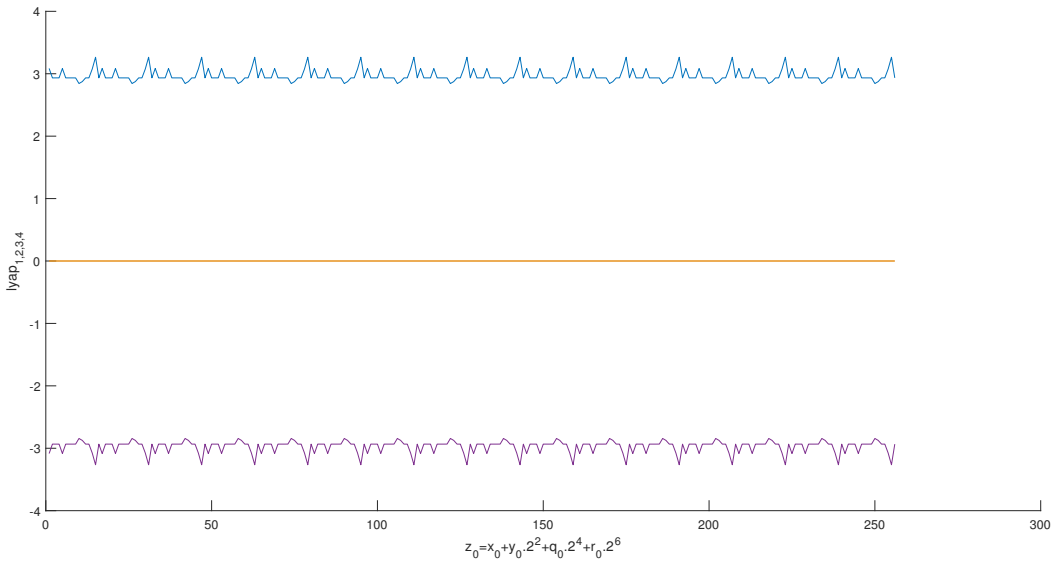


FIGURE 2.5 – Evolution des exposants de lyapunov sur 2 bits de la 4D-PWLCM en fonction de z_0

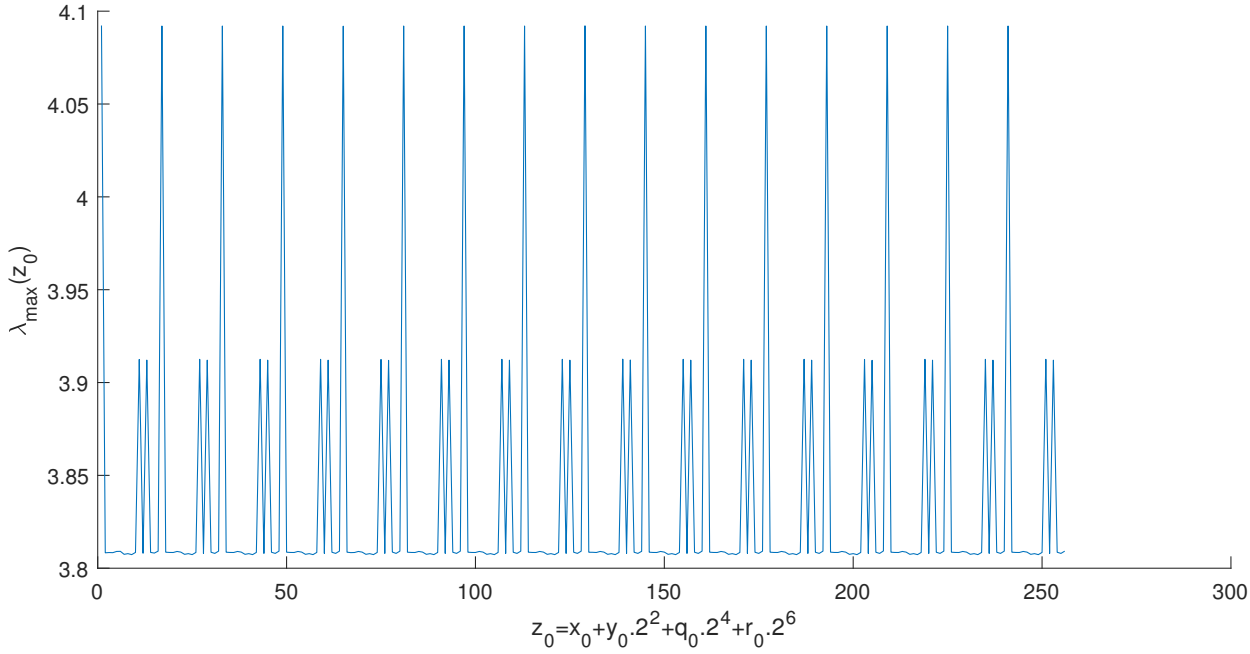


FIGURE 2.6 – Evolution du plus grand exposant de lyapunov sur 2 bits de la 4D-PWLCM en fonction de z_0

tique car parmi les exposants nous avons un qui est strictement positif. La somme de ses exposants est également nulle, donc le système est conservatif.

Nous pouvons aussi l'utiliser pour une application de chiffrement d'images, mais dans ce cas nous ferons des regroupements de blocs d'image de taille 4×4.

L'implémentation d'un algorithme de chiffrement sur 2 bits peut être faite sur les circuits élec-

troniques de petite précision comme les capteurs, certaines familles de microcontrolleurs, dans les mini-ordinateurs etc.

Dans ce travail de thèse, nous présenterons l'algorithme de chiffrement d'images en utilisant la 4-D PWLCM sur 2 bits. Nous présenterons l'algorithme détaillée sur 4 bits, ensuite, on s'inspirera de cet algorithme pour implémenter celui de 2 bits. Ceci donc fera l'objet du chapitre 3 que nous présenterons .

D'après la figure (2.5), nous observons que le plus grand exposant de Lyapunov reste positif pour différentes combinaisons de conditions initiales x_0,y_0, q_0 et r_0 prises dans l'espace des phases. Ainsi, sur 2 bits, la 4-D PWLCM conserve aussi une dynamique chaotique. Pour effectuer la simulation, nous avons pris 4 termes de pertubations (paramètres de contrôles) sur chaque variable d'état de la 4D-PWLCM. Ces valeurs sont données par : $a = \{1,1,0,1,2\}$; $e = a$; $c = \{0,3,1,2,3\}$; $g = c$; $b = \{0,2,0,1,1\}$; $f = b$; $d = \{3,1,1,2,0\}$; $h = d$;

b) Exposants de Lyapunov en fonction des paramètres de contrôle

Comme nous l'avons fait dans la cadre de 4 bits, nous appliquons également le même principe en utilisant une précision de 2 bits.

Pour cela, faisons varier les paramètres de contrôles **b** cette fois ci, en utilisant cette fois ci 4 termes de pertubations (b_1, b_2 , b_3 et b_4).

L'équation traduisant la modification des valeurs de ces paramètres de contrôles dans l'espace de phases est donnée par :

$$b_r = \sum_{j=1}^{N} 2^{n(j-1)} b_j \tag{2.29}$$

où, $n = 2$ (le nombre de bits) ; $N = 4$ (nombre de termes de pertubation).

Ainsi, la représentation des exposants de Lyapunov en fonction des variations des paramètres de contrôles b_1, b_2 , b_3, et b_4 est donnée sur le graphe ci-dessous.

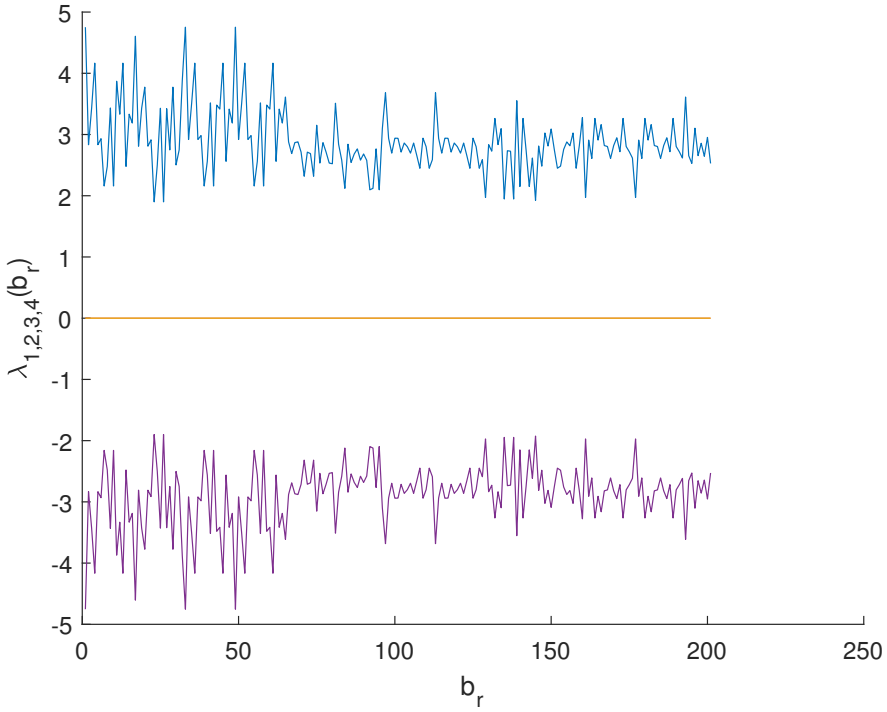


FIGURE 2.7 – Evolution des exposants de lyapunov sur 2 bits de la 4D-PWLCM en fonction des variations du paramètre de contrôle **b**

2.4) Périodes de la récurrence d'Arnold 4-D proposée

2.4.1) Méthode de calcul des périodes

Soit le système modifié d'Arnold en 2-D définit par morceaux donnée par [] :

$$\begin{cases} x(n+1) = x(n) + \alpha y(n) + \sum_{i=1}^{M} (a_i + y(n)) \bmod c_i \\ y(n+1) = \beta x(n+1) + y(n) + \sum_{j=1}^{P} \left(b_j + x(n+1)\right) \bmod d_j \end{cases} \bmod N$$

Pour effectuer le calcul, on fixe tout d'abord la précision de calcul $N = 2^n$, où n représente ici le nombre de bits. Nous prendrons d'abord le cas où $n = 2$ pour effectuer le calcul ; ensuite nous allons étendre les calculs sur le nombre de bits c'est à dire, $n = 3,4,5,6,7$et 8.

Prenons le cas $n = 2$; les conditions initiales x_0 et y_0 sont prises de 0 à $2^2 - 1$, c'est à dire $x_0 \in \{0,1,2,3\}$ et $y_0 \in \{0,1,2,3\}$. En effectuant les calculs sur la récurrence d'Arnold simple présenté dans la littérature, il convient de choisir les paramètres de contrôles suivant : $\alpha = \beta = 1$; $M = P = 1$; $a_1 = b_1 = 2$; $c_1 = d_1 = 1$;

Ainsi, les résultats obtenus sur le logiciel Matlab R2018b des séquences x et y données pour chaque conditions initiales x_0 et y_0 peuvent être regroupés dans le tableau ci-dessous.

x0/y0	0	1	2	3
0	x=[0,0,0,0,0,0,0] y=[0,0,0,0,0,0,0]	x= [0,1,3,0,1,3,0] y=[1,2,1,1,1,1,2]	x=[0,2,2,0,2,2,0] y=[2,0,2,2,0,2,2]	x=[0,3,1,0,3,1,0] y=[3,2,3,3,2,3,3]
1	x=[1,1,2,1,1,2,1] y= [1,3,0,1,3,0,1]	x=[1,2,1,1,2,1,1] y=[1,3,0,1,3,0,1]	x=[1,3,0,1,3,0,1] y=[2,1,1,2,1,1,2]	x=[1,0,3,1,0,3,1] x=[3,3,2,3,3,2,3]
2	x=[2,2,0,2,2,0,2] y= [0,2,2,0,2,2,0]	[2,3,3,2,3,3,2] [1,0,3,1,0,3,1]	[2,0,2,2,0,2,2] [2,2,0,2,2,0,2]	[2,1,1,2,1,1,2] [3,0,1,3,0,1,3]
3	[3,3,2,3,3,2,3] [0,3,1,0,3,1,0]	[3,0,1,3,0,1,3] [1,1,2,1,1,2,1]	[3,1,0,3,1,0,3] [2,3,3,2,3,3,2]	[3,2,3,3,2,3,3] [3,1,0,3,1,0,3]

TABLE 2.1 – calcul des séquences x et y après 7 itérations pour différentes conditions initiales x_0 et y_0

Après avoir calculé les séquences x et y, on cherche ensuite la valeur d'itération (indice i) qui obéit simultanément aux équations suivantes :

$$x(i+1) - x_0 = 0 \tag{2.30}$$

et

$$y(i+1) - y_0 = 0 \tag{2.31}$$

Où, x_0 et y_0 sont les valeurs de x et y à l'instant initial.
Ainsi, en utilisant ces équations et le tableau (2.1), nous recherchons donc les indices d'itérations de coïncidence en x et en y. Ce qui nous permet d'avoir le tableau suivant.

x0/y0	0	1	2	3
0	$x(2)-x(1)=0$ $y(2)-y(1)=0$ $i=1$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$
1	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$
2	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$
3	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$	$x(4)-x(1)=0$ $y(4)-y(1)=0$ $i=3$

TABLE 2.2 – Indices de coïncidence de x et y

Ainsi, les périodes correspondantes sont donc ces indices enregistrés dans le tableau ci-dessous. Pour avoir la période du système complet, il suffit de calculer le plus petit commun multiple ($ppcm$) des périodes individuelles prise pour chaque condition initiale. Ainsi en calculant le ppcm des indices (périodes individuelles) enregistrés dans le tableau ci-dessous, nous obtenons :

$$T = ppcm(T_{11}(0,0), T_{12}(0,1) + ... + T_{44}(3,3)) \tag{2.32}$$

Ainsi, par application de l'équation ci-dessous, on calcule la période :
$T = ppcm(1,3,3,3,...,3) = 3$

2.4.2) *Calcul des périodes de la récurrence 2D et 4D*

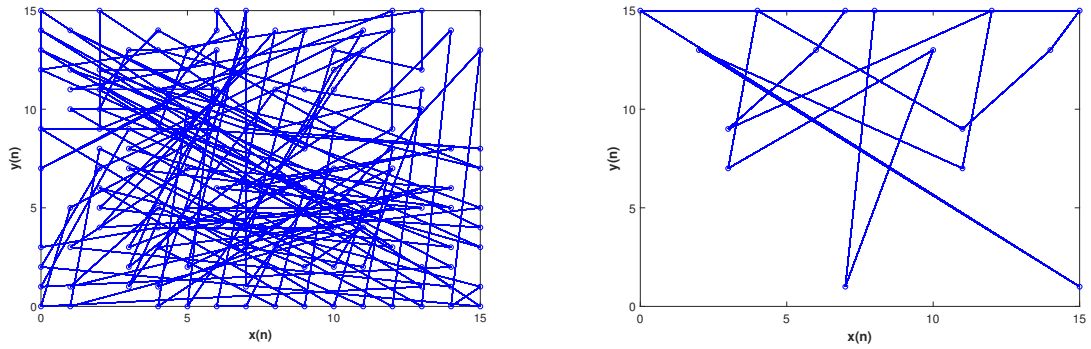
2.4.2.1) *Périodes de la récurrence d'Arnold 2-D*

En utilisant la méthode de calcul des périodes présentée ci-dessous, on calcule les périodes de la récurrence d'Arnold conventionnel (ACM) et celle de la récurrence 2-D PWLCM définit dans [] . Les valeurs de ces périodes sont regroupées dans le tableau ci-dessous en tenant compte des différentes combinaisons de paramètres de contrôles .
D'après le tableau ci-dessous, nous remarquons que, les périodes de la récurrence d'Arnold 2-D

n	(a_1, b_1)	(c_1, d_1)	T_{ACM}	$T_{2D-PWLCM}$
1	(1,2)	(2,3)	3	4
2	(1,2)	(3,3)	3	90
3	(3,3)	(2,3)	6	780
4	(5,1)	(11,3)	12	12759390
5	(2,2)	(7,5)	24	56934108
6	(6,2)	(5,13)	48	$1.7870e+12$
7	(2,2)	(7,5)	96	$1.2041e+16$
8	(6,2)	(5,13)	192	$6.918e+56$

TABLE 2.3 – Calcul des périodes de la récurrence ACM et 2-D PWLCM modifiée

modifiée par morceaux sont beaucoup plus grande que celle de la récurrence d'Arnold conventionnelle. Ceci donc nous conduit à utiliser la récurrence d'Arnold 2-D modifiée comme outil pour la conception de notre cryptosystème, car elle génère des séquences pseudo-aléatoires. Nous pouvons observer donc celà sur le portrait de phase donné à la figure ci-dessous.



(a) (b)

FIGURE 2.8 – Portrait de phase de la récurrence d'Arnold : a) Récurrence d'Arnold linéaire par morceaux 4D (RALM 4D) ; b) Récurrence d'Arnold (ACM)

2.4.2.2) *Périodes de la récurrence d'Arnold 4-D*

Pour calculer les périodes de la récurrence 4-D, on utilise l'extension de la récurrence 2-D en 4-D donnée par le système (2.7) :

$$\begin{cases} x(n+1) = x(n) + \alpha y(n) + F_1(y,n) \\ y(n+1) = y(n) + \beta x(n+1) + F_2(x,n) \\ z(n+1) = z(n) + \gamma y(n+1) + F_3(y,n) \\ t(n+1) = t(n) + \zeta z(n+1) + F_4(z,n) \end{cases} \text{mod } N$$

où, $F_1(y,n)$,$F_2(x,n)$, $F_3(y,n)$ et $F_4(z,n)$ sont les termes de pertubations non linéaires apportés sur chaque variable d'état du système. Les expressions mathématiques de ces termes sont donnés précédemment.

En suivant la procédure de calcul des périodes définit plus haut dans le cas de la récurrence 2-D, on déduit celle de la récurrence 4-D et le tableau donnant les résultats du calcul des périodes pour différentes conditions initiales sont reportés dans le tableau ci-dessous.

Nous observons que les périodes obtenus dans le cas de la récurrence 4-D sont beaucoup plus

n	(a_1,b_1,e_1,f_1)	(c_1,d_1,g_1,h_1)	T_{4D-ACM}	$T_{4D-PWLCM}$
1	(1,0,0,1)	(1,1,0,0)	6	6
2	(1,2,1,0)	(3,3,3,3)	12	360
3	(6,3,1,1)	(11,1,5,11)	24	240240
4	(6,3,1,1)	(7,1,1,1)	48	340728960

TABLE 2.4 – Calcul des périodes de la récurrence 4D-ACM et 4-D PWLCM modifiée

grandes que celle de la récurrence 2-D ; ce qui justifie donc l'intérêt d'utiliser ce système pour concevoir un algorithme de chiffrement.

2.5) Logiciel d'implémentation

Pour la conception des algorithmes de chiffrements, le logiciel que nous avons utilisé est MATLAB 2018b.

Partout dans le monde, des millions d'ingénieurs et de scientifiques utilisent MATLAB pour analyser et concevoir les systèmes et produits de demain. MATLAB est présent dans des systèmes automobiles de sécurité active, des véhicules spatiaux, des appareils de surveillance médicale, des réseaux électriques intelligents et des réseaux mobiles LTE. Il est utilisé dans les domaines de l'apprentissage automatique, le traitement du signal, la vision par ordinateur, les communications, la finance computationnelle, la conception de contrôleurs, la robotique et bien plus.

L'interface du logiciel se présente comme illustré à la figure ci-dessous.

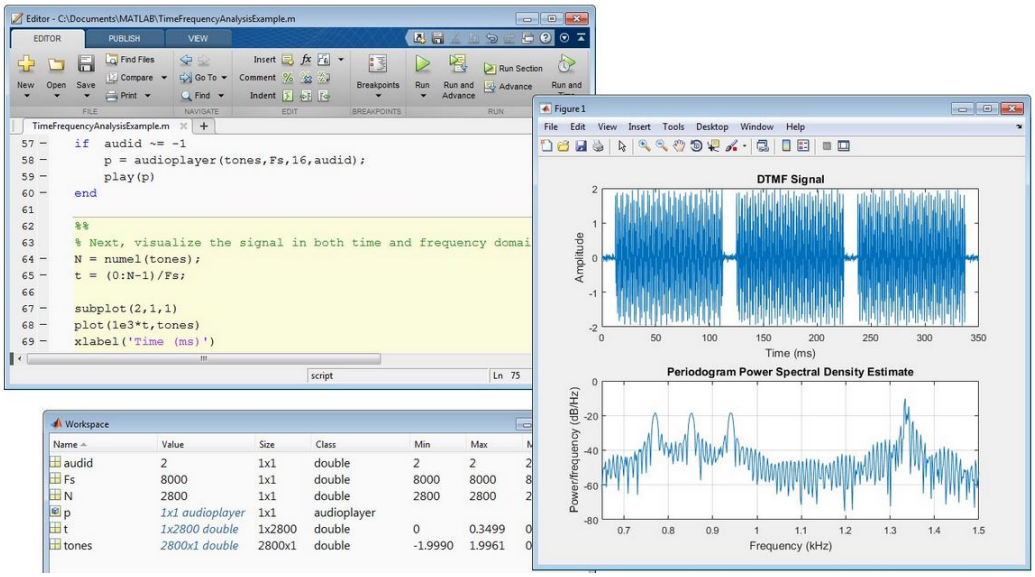


FIGURE 2.9 – Présentation de l'interface du logiciel Matlab

Les données que nous utiliserons pour la conception de l'algorithme de chiffrement sont les images. Les images que nous utiliserons sont les images couleurs où en niveau de gris. Les images doivent etre de taille carrée c'est à dire le nombre de lignes et le nombre de colonnes sont équivalents. Dans où, on est dans le cas d'une image non carrée, il faudrait faire du bourrage avec des zéros afin d'obtenir une taille carrée.

Les images fournies par les appareils photo sont généralement en couleur. Une image est dite

couleur lorsqu'elle est constituée de 3 couches à savoir, le rouge, le vert et le bleu en abrégé RGB. Chaque couche est une matrice comportant Ny lignes et Nx colonnes. Le plus souvent, cette matrice contient des entiers codés sur 8 bits (les valeurs vont de 0 à 255).Pour l'image en couleur complète, il y a donc 24 bits par pixels, à multiplier par le nombre de pixels pour obtenir l'occupation totale en mémoire. Chaque couche peut être vue comme une image en niveaux de gris. Le niveau 0 est le noir, le niveau 255 est le blanc, le niveau 128 est un gris moyen.
Nous avons donc ci-dessous la base d'images à utiliser pour effectuer les tests sur l'algorithme de chiffrement sur 4 bits et ensuite sur le chiffrement sur 2 bits.
La base est constituée de 2 familles d'images à savoir les images couleurs et celles en niveaux de gris. Ces images sont les images tests standards utilisés généralement pour faire les tests sur les algorithmes de chiffrement proposés dans la littérature. Il est aussi à noter que la tailles de ces images peut varier. Nous avons des images de dimensions 256×256, 512×512, 1024×1024 etc. Certaines parmi elles peuvent etre aussi de dimension rectangulaire c'est à dire le nombre de lignes différent au nombre de colonnes, et dans ce cas, on effectuera une opération qui est faite dans le traitement d'image appelée le bourrage des 0 (appelée en Anglais 0-padding).

(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

FIGURE 2.10 – Images en couleur et niveaux de gris : (a) Lena ; (b) Barbara ; (c) boat en niveaux de gris ; (d) Peppers en niveaux de gris ; (e) Bird en gris ; (f) bird ; (g)Peppers ; (h) Baboon ; (i) Lena en gris

Conclusion

Dans ce chapitre, il était question de présenter l'étude analytique de la récurrence linéaire par morceaux d'Arnold 4-D respectivement sur 4 bits et 2 bits. Il vient de cet étude, que nous pouvons utiliser celles-ci pour l'élaboration d'un algorithme de chiffrement d'images . Elles sont chaotiques, possèdent des séquences pseudo-aléatoires lors du calcul des périodes pour certaines valeurs de paramètres de controles. Il est convenu aussi de remarquer, que les séquences sur 4 bits présentent plus d'aléa que celles sur 2 bits lorsqu'on observe le tableau des périodes. Le chiffrement sur 2 bits porte plus d'avantage dans la mesure où elle réduit le temps de calcul , car elle nécessitera moins d'opérations que celle sur 4 bits. Les compléments justi-

ficatifs de choix de ces 2 récurrences dans cette thèse, seront plus explicite dans le chapitre suivant.

CHAPITRE 3

RÉSULTATS ET DISCUSSIONS

Introduction

Les résultats analytiques présentés au chapitre 2, nous ont permis d'observer que, la récurrence d'Arnold est un outil de base pour le brouillage des données. En apportant des termes de pertubations sur la récurrence 2-D, on observe une dynamique quasi non-régulière avec une période infiniment grande. Ceci nous conduit donc à l'utiliser comme outil pour la conception d'un algorithme de chiffrement . Dans ce chapitre, nous allons présenter une application cryptographique à base d'images en utilisant la récurrence d'Arnold en dimension 4 codée sur 4 bits. Nous présenterons d'une part l'algorithme de chiffrement, et d'autre part, l'algorithme de déchiffrement en utilisant le logiciel Matlab 2018b. Enfin, pour prouver l'efficacité de notre algorithme, nous allons l'évaluer à l'aide des métriques utilisées en cryptographie.

3.1) Présentation de l'algorithme sur 4 bits

Comme nous l'avons dit précédemment, le système que nous allons utiliser est la récurrence 4-D, car elle permet d'effectuer les opérations de confusion et de diffusion en une seule étape. Ceci rend l'algorithme rapide, flexible et simple à utiliser. La méthode de chiffrement que nous allons utiliser est le chiffrement par bloc car celle ci est beaucoup plus simple pour la mettre en œuvre. Après la phase de diffusion et de confusion avec la récurrence linéaire d'Arnold par morceaux 4-D (4-D PWLCM), on utilise la récurrence linéaire d'Arnold par morceaux 2-D (2-D PWLCM) pour effectuer à nouveau une confusion supplémentaire afin d'accroitre la robustesse de notre algorithme ainsi que la taille de la clef de chiffrement. Les conditions initiales

et les paramètres de contrôles de la 4D-PWLCM proviennent de la clef externe de sécurité tandis que les paramètres de contrôles de la 2D-PWLCM proviennent des valeurs produites par la 4D-PWLCM. Ainsi, ceci implique que la 2-D PWLCM effectuant la confusion dépend de l'image permutée et diffusée provenant de la 4-D PWLCM.

3.1.1) Algorithme utilisant la 4D-PWLCM et 2D-PWLCM sur 4 bits

Les différentes étapes pour l'élaboration de l'algorithme de chiffrement des données images sont les suivantes :

1) Considérer une image de taille $N_L \times N_C$ et la vectoriser en une dimension pour avoir une image en 1D.

2) Diviser l'image 1-D obtenu en blocs de taille $1 \times 2^4 * 2^4$ qui correspondant aux blocs de taille 16×16.

3) A partir d'une clef externe fixée, on détermine les paramètres de contrôles **a**, **b**,**c**, **d**, **e**,**f**,**g**, **h**.

4) Décomposer les valeurs de chaque pixels de l'image en entrée, en 4 bits pour les variables d'états x et y respectivement qui effectue la confusion des pixels . Ensuite pour effectuer l'opération de diffusion on utilise les variables q et r également décomposés sur 4 bits respectivement.

5) Appliquez la récurrence linéaire d'Arnold 4-D par morceaux sur chacune des pixels de l'image de départ en utilisant les valeurs de x,y,q et r décomposées sur 4 bits respectivement.

6) Répéter les étapes 5 et 6 jusqu'à ce que l'image d'un bloc soit modifié entièrement. Une fois le bloc de 16×16 modifié, on considère le bloc suivant.

7) Répéter les étapes 5 et 6 afin d'obtenir l'image chiffrée reconstitué à partir des blocs de 16×16.

8) Déterminer les paramètres de contrôles de la récurrence d'Arnold 2D-PWLCM à partir le l'image chiffrée obtenu à l'étape précédente.

8) Effectuer la permutation des blocs de l'image chiffrée en utilisant la récurrence 2-D PWLCM codé sur 8 bits car les valeurs de pixels de l'image sont codées sur 8 bits.

9) L'image obtenue après la confusion des blocs est donc l'image cryptée après un tour . Pour effectuer un second tour, on remonte à l'étape 4 et ainsi de suite jusqu'à l'obtention d'une image totalement cryptée.

3.1.2) Principe de distribution de la clef externe

La clef externe est définit en utilisant N_k caractères ASCII, avec $0 \leq k \leq N_k - 1$. Pour une meilleure sécurité dans un algorithme de chiffrement, la clef doit être de taille minimale de 256 bits. Pour avoir la taille vectorielle κ des paramètres de contrôles, **a**,**b**,**c**,**d**,**e**,**f**,**g**,**h**, on effectue l'opération suivante :

$$\kappa = 2\lfloor \frac{N_k}{8} \rfloor \tag{3.1}$$

La relation ci-dessous est utilisée parce que, un caractère ASCII est codée sur 8 bits .
Pour déterminer le nombre de caractères ASCII correspondant pour chaque vecteur de paramètre de contrôles, on utilise la relation suivante :

$$\theta = \frac{\kappa}{2}\, ASCII \tag{3.2}$$

Ainsi, pour une clef de taille 256 bits, les paramètres de contrôles sont prises comme :

$$\begin{cases} \mathbf{a} = (a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8) \\ \mathbf{b} = (b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8) \\ \mathbf{c} = (c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8) \\ \mathbf{d} = (d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8) \\ \mathbf{e} = (e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8) \\ \mathbf{f} = (f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8) \\ \mathbf{g} = (g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8) \\ \mathbf{h} = (h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8) \end{cases} \tag{3.3}$$

Chaque caractère ASCII est scindé en 2 blocs de 4 bits chacune et représentant respectivement les coordonnées des vecteurs de paramètres de contrôles définit sur le système ci-dessous.
Soit une clef représenté sur 256 bits donné par les caractères suivants :

$Clef = C_0C_1C_2C_3C_4C_5C_6C_7C_8C_9C_{10}C_{11}C_{12}C_{13}C_{14}C_{15}C_{16}C_{17}C_{18}C_{19}C_{20}$
$C_{21}C_{22}C_{23}C_{24}C_{25}C_{26}C_{27}C_{28}C_{29}C_{30}C_{31}$.

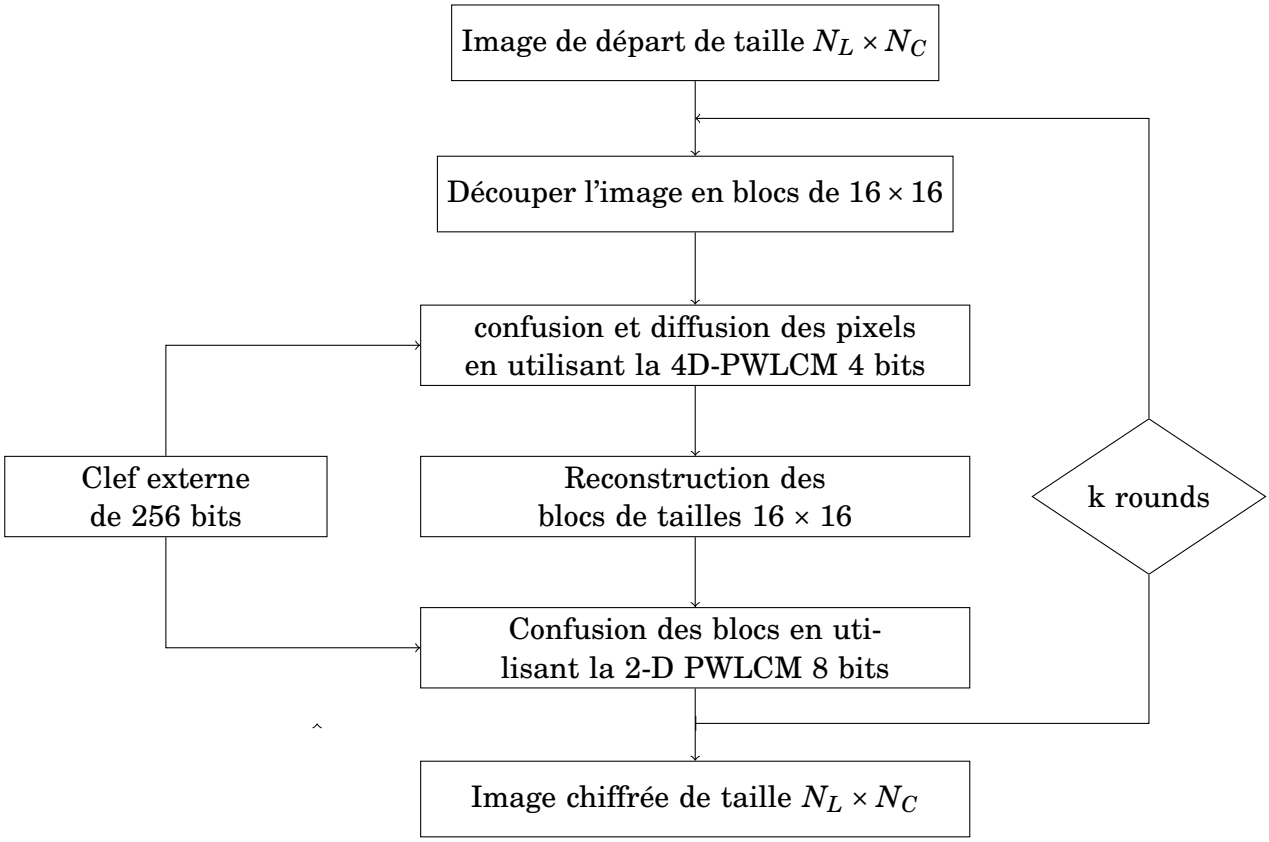


FIGURE 3.1 – Organigrame de chiffrement utilisant la 4-D PWLCM sur 4 bits

Pour représenter les coordonnées des vecteurs de paramètres de contrôles **a**, on prend les caractères C_0 à C_3, pour **b**, C_4 à C_7, pour **c**, C_8 à C_{11}, **d**, C_{12} à C_{15}, **e**, C_{16} à C_{19}, **f**, C_{20} à C_{23}, **g**, C_{24} à C_{27} et **h**, C_{28} à C_{31} .

Dans le cas d'une clef de taille 2048 bits par exemple, on utilisera 256 caractères ASCII, car un caractère ASCII est représenté sur 8 bits. En utilisant la relation (3.1) ci-dessous, on aura un vecteur de taille $\kappa = 64$ éléments de paramètres de contrôles. Chaque élément du vecteur de paramètre de contrôle étant codé sur 4 bits . Ainsi, en utilisant la relation (3.2), on aura besoin de $\theta = 32$ caractères ASCII pour représenter chaque vecteur de paramètre de contrôle **a**, **b**, **c**, **d**, **e**, **f**, **g** et **h**.

Les expressions mathématiques permettant de donner les composantes du vecteur de paramètres **a** peuvent être définis comme :

$$\begin{cases} \mathbf{a}(2\xi-1) = \frac{C_{2\xi-1}}{16} \\ \\ \mathbf{a}(2\xi) = C_{2\xi-1}\ mod 16 \end{cases} \tag{3.4}$$

Pour ceux du vecteur **c**, on utilise :

$$\begin{cases} \mathbf{c}(2\xi-1) = \mathbf{a}(2\xi-1) + \lfloor \frac{C_{\xi+\kappa-1}}{16} \rfloor \\ \\ \mathbf{c}(2\xi) = \mathbf{a}(2\xi) + (C_{\xi+\kappa-1}\ mod 16) \end{cases} \tag{3.5}$$

où, $1 \leq \xi \leq \theta$ et C_i représentant les différents caractères ASCII. On effectue la même procédure pour déterminer les composantes respectives des vecteurs **b**, **d**, **e**, **g**, **f** et **h**.

3.1.3) Processus de permutation et de diffusion

En s'inspirant de l'étape 4, on décompose individuellement chaque pixel P_i d'une sous image notée S_j ($j \in \mathbb{N}$) en 2 entiers naturels q_i et r_i codés respectivement sur 4 bits. Ainsi, comme une pixel est codée sur 8 bits, les entiers q_i et r_i sont donnés par :

$$\begin{cases} q_i = \lfloor \frac{P_i}{16} \rfloor \\ \\ r_i = P_i\ mod 16 \end{cases} \tag{3.6}$$

Comme nous l'avons dit précédemment, les opérations de confusion et de diffusion s'effectuent en une seule étape en utilisant la récurrence 4D-PWLCM proposé. En effet, chaque pixel d'intensité q_i et r_i avec pour coordonnées x_i et y_i est utilisée comme condition initiale pour la récurrence 4D-PWLCM. Après quelques itérations de la récurrence 4D-PWLCM définit en (2.7), on obtient des nouvelles valeurs de coordonnées x'_i, y'_i, q'_i et r'_i .
A chaque sous image S_j correspond une combinaison , **a**(k), **b**(k), **c**(k),**d**(k), **e**(k), **f**(k) et **g**(k) de paramètres de contrôles avec $k > 0$.
Ainsi, pour chaque bloc image S_j ($j \in \mathbb{N}$) le système 4D-PWLCM se présentera ainsi comme :

$$\begin{cases} x(t+1) = x(t) + \alpha y(t) + (\mathbf{a}(k) + y(t)) \mod \mathbf{c}(k) \\ y(t+1) = y(t) + \beta x(t+1) + (\mathbf{b}(k) + x(t+1)) \mod \mathbf{d}(k) \\ q(t+1) = q(t) + \gamma y(t+1) + (\mathbf{e}(k) + y(t+1)) \mod \mathbf{g}(k) \\ r(t+1) = r(t) + \zeta q(t+1) + (\mathbf{f}(k) + q(t+1)) \mod \mathbf{h}(k) \end{cases} \mod 16 \tag{3.7}$$

où, $k = 1 + j \ mod \kappa$.

La nouvelle position i' de la pixel diffusée est obtenue après 3 itérations de la récurrence 4D-PWLCM est donnée par :

$$i' = 16.y_{i'} + x_{i'} \tag{3.8}$$

La valeur de la pixel correspondante à la nouvelle position i' est donnée par :

$$P_{i'} = 16.q_{i'} + r_{i'} \tag{3.9}$$

Pour augmenter la sécurité de notre algorithme proposé, on effectue une autre opération de permutations des pixels de l'image reconstituée obtenu après l'étape 7. L'opération de permutation s'effectue en utilisant le 2D-PWLCM définit comme :

$$\begin{cases} x(t+1) = (x(t) + \alpha y(t) + \sum_{k=1}^{16} (\mathbf{a_1}(k) + y(t)) \mod \mathbf{c_1}(k)) \mod m_1 \\ y(t+1) = (\beta x(t+1) + y(t) + \sum_{k=1}^{16} (\mathbf{b_1}(k) + x(t+1)) \mod \mathbf{d_1}(k)) \mod m_2 \end{cases} \tag{3.10}$$

Les paramètres de contrôles, $\mathbf{a_1}$, $\mathbf{b_1}$, $\mathbf{c_1}$ et $\mathbf{d_1}$ contiennent respectivement 16 éléments dont chacun d'eux est codé respectivement sur 4 bits. Ces éléments proviennent d'une part, de la clef externe du chiffrement et d'autre part, de l'image permutée et diffusée issue de la 7ème étape de l'algorithme ; En considérant, 32 caractères ASCII, permettant de représenter chaque vecteur, et provenant de la clef externe, ces coordonnées de paramètres de contrôles peuvent s'écrire comme :

$$\begin{cases} \mathbf{a_1}(1:\kappa) = \Gamma \bmod \mathbf{c} \\ \mathbf{a_1}(\kappa+1:2\kappa) = \Gamma \bmod \mathbf{d} \\ \mathbf{b_1}(1:\kappa) = \Gamma \bmod \mathbf{g} \\ \mathbf{b_1}(\kappa+1:2\kappa) = \Gamma \bmod \mathbf{h} \\ \mathbf{c_1}(1:\kappa) = \mathbf{a_1}(1:\kappa) + \Gamma \bmod \mathbf{a} \\ \mathbf{c_1}(\kappa+1:2\kappa) = \mathbf{a_1}(\kappa+1:2\kappa) + \Gamma \bmod \mathbf{b} \\ \mathbf{d_1}(1:\kappa) = \mathbf{b_1}(1:\kappa) + \Gamma \bmod \mathbf{e} \\ \mathbf{d_1}(\kappa+1:2\kappa) = \mathbf{b_1}(\kappa+1:2\kappa) + \Gamma \bmod \mathbf{f} \end{cases} \tag{3.11}$$

Avec

$$\Gamma = \sum_{i=1}^{N_L} \sum_{j=1}^{N_C} I_c(i,j). \tag{3.12}$$

où, I_c est l'image permutée et diffusée en utilisant la 4D-PWLCM .

Les constantes m_1 et m_2 sont définies tel que :

$$\begin{cases} m_1 = \frac{N_L}{T_1} \\ m_2 = \frac{N_C}{T_2} \end{cases} \tag{3.13}$$

Avec, $(T_1, T_2) \in \mathbb{N}^2_{\geq} 1$. $T_1 \times T_2$ est la taille des sous images où des images blocs c'est à dire, $T_1 = 16$ et $T_2 = 16$. $N_L \times N_C$ est la taille de l'image de départ où, l'image d'entrée de notre algorithme. Ces images peuvent être de taille 256×256, 512×512, 1024×1024, etc...

Généralement, ces images en entrées sont de taille carrée c'est à dire le nombre de lignes est égal au nombre de colonnes ($N_L = N_C$). Ces images peuvent être en couleur où en niveau de gris.

3.2) Résultats et analyse de sécurité sur 4 bits

Les images utilisées pour effectuer les tests de notre algorithme sont les images standards données par : "Lena", "Baboon", "Peppers", de taille 512×512 , avec 256 niveaux de gris. La figure ci dessous, illustre les différentes images chiffrées et déchiffrées de ces images standards en utilisant une même clef de sécurité.

Les simulations ont été effectuées sur le logiciel Matlab 2018b en utilisant une machine ayant les caractéristiques suivantes : i5-8250u CPU@ 1.60 Ghz, 8GB de RAM, et Windows 10 comme système d'exploitation.

Les simulations dans ce cas sont effectuées avec une clef externe de 256 bits notée $K_1 = azertyuiopqsdfgjazertyuiopqsdfg0$

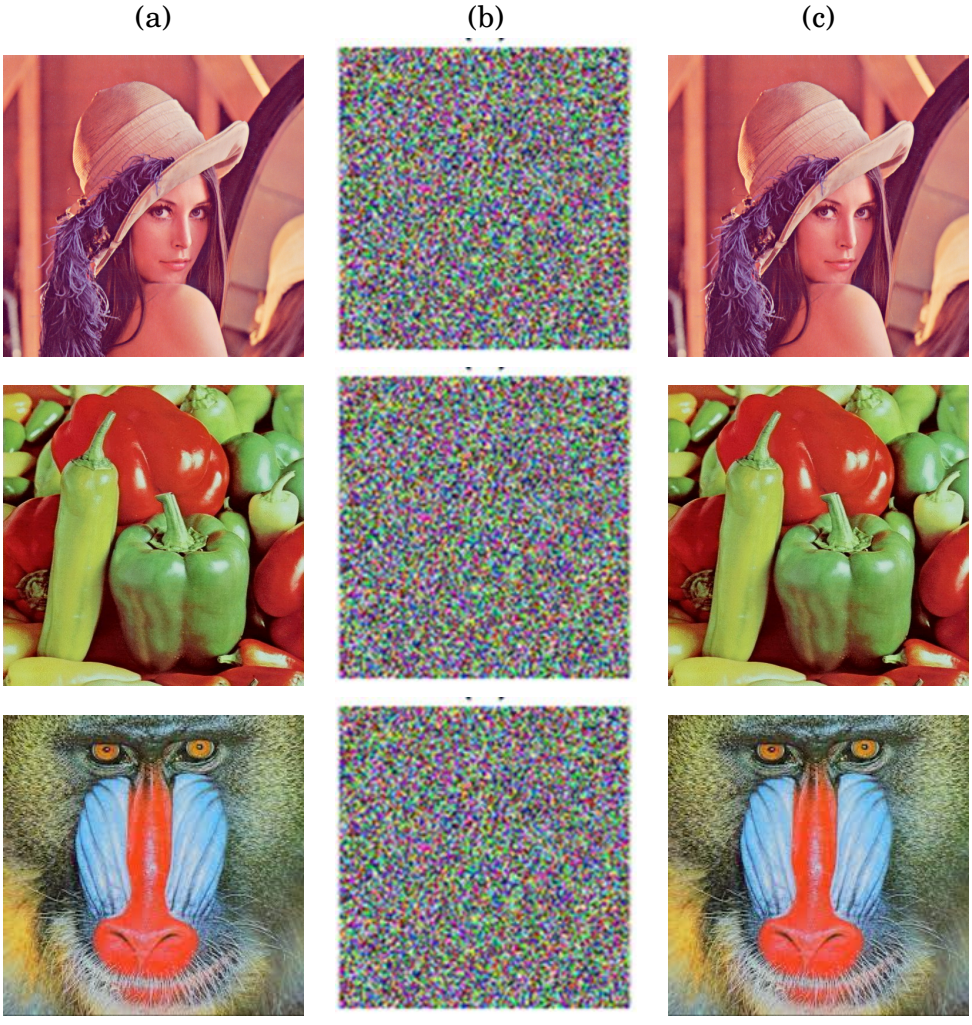


FIGURE 3.2 – Quelques exemples de chiffrement d'images de Lena, Peppers et Baboon : a) Images originale, b) Images chiffrées, c) Images déchiffrées

3.2.1) Analyse des métriques

3.2.1.1) Calcul de l'entropie

L'entropie de Shannon est un indicateur majeur permettant de confirmer que l'image chiffrée est sécurisée où pas, en effectuant différentes permutations de pixels sur celle-ci. Il permet de mesurer le désordre où le degré d'aléa des différentes permutations de pixels contenues dans une image cryptée où chiffrée.

Pour une image codée sur 8 bits, l'expression du calcul de l'entropie est donnée par :

$$H = -\sum_{i=0}^{255} p(v_i) log_2(p(v_i)) \tag{3.14}$$

où, $0 \leq v_i \leq 255$ représente les valeurs de pixels et $p(v_i)$, la probabilité d'apparition de la pixel v_i. La valeur idéale pour le calcul de l'entropie d'une image chiffrée est $H = 8$.

Les résultats obtenus pour le chiffrement des images en utilisant la clef de 256 bits et 2048 bits sont regroupés dans le tableau ci-dessous.

TABLE 3.1 – Résultats sur les calculs de l'entropie

Images	Couleurs	Entropies (256 bits)	Entropies (2048 bits)
Lena	Rouge	7.9994	7.9994
	Verte	7.9994	7.9991
	Bleue	7.9993	7.9993
Baboon	Rouge	7.9994	7.9994
	Verte	7.9992	7.9992
	Bleue	7.9992	7.9993
Airplane	Rouge	7.9993	7.9992
	Verte	7.9993	7.9992
	Bleue	7.9993	7.9994
Peppers	Rouge	7.9993	7.9993
	Verte	7.9992	7.9993
	Bleue	7.9988	7.9993

3.2.1.2) Calcul du coefficient de corrélation

En général, la corrélation entre les pixels adjacentes d'une image naturelle est en majorité toujours grande (proche de 1). Ainsi, la corrélation des pixels adjacentes d'une image cryptée, est l'un des critères important pour évaluer la performance d'un algorithme de chiffrement.

Pour calculer le coefficient de corrélation, on utilise la relation suivante [90] :

$$\rho = \frac{\sum_{i=1}^{M}(x_i - E(x))(y_i - E(y))}{\sqrt{\sum_{i=1}^{M}(x_i - E(x))^2 \times \sum_{i=1}^{M}(y_i - E(y))^2}} \tag{3.15}$$

où, $E(x) = \frac{1}{M}\sum_{i=1}^{M} x_i$ et $E(y) = \frac{1}{M}\sum_{i=1}^{M} y_i$.
Les variables x et y sont les valeurs de pixels en niveau de gris, M étant le nombre total de pixels contenu dans une image .
Pour un bon test d'un algorithme de chiffrement, on doit avoir la valeur ρ d'une image cryptée très faible (proche de la valeur 0).
Les résultats de calcul des coefficients de corrélation des images chiffrées en utilisant respectivement les clefs de 256 bits et 2048 bits sont regroupés dans le tableau suivant.

TABLE 3.2 – Résultats des tests de coefficients de corrélation

Images	Couleurs	ρ (256 bits)	ρ (2048 bits)
Lena	Rouge	0.0017	0.0004
	Verte	−0.0015	0.0000
	Bleue	−0.0010	−0.0043
Baboon	Rouge	0.0012	0.0004
	Verte	0.0006	−0.0010
	Bleue	7.9992	0.0017
Airplane	Rouge	7.9993	0.0004
	Verte	7.9993	0.0008
	Bleue	7.9993	0.0005
Peppers	Rouge	−0.0017	−0.0027
	Verte	0.0028	−0.0033
	Bleue	0.0039	0.0006

3.2.1.3) *Analyse des attaques différentiels*

L'attaque différentielle a pour but de chiffrer l'image en clair, en apportant une légère modification sur celle-ci, puis de trouver les différences entre les 2 images chiffrées.
En cryptographie de données, deux paramètres importants, connus sous les noms de UACI (Unified Average Changing Intensity) et NPCR (Number of Pixels Change Rate) [] sont utilisés pour évaluer les performances des algorithmes de chiffrement face aux différentes attaques.

Pour une image de 256 niveaux de gris, le NPCR et le UACI entre deux images notées A et B se calcule en utilisant les équations suivantes :

$$NPCR_{A,B} = \frac{\sum_{i=1}^{NL}\sum_{j=1}^{NC} D(i,j)}{N_L \times N_C} \times 100 \tag{3.16}$$

où,

$$D(i,j) = \begin{cases} 1, \text{ si } \quad A(i,j) \neq B(i,j) \\ 0 \text{ ailleurs .} \end{cases} \tag{3.17}$$

$$UACI_{A,B} = \frac{100}{255} \frac{\sum_{i=1}^{N_L}\sum_{j=1}^{N_C} |A(i,j) - B(i,j)|}{N_L \times N_C} \tag{3.18}$$

Avec, N_L et N_C représentant le nombre de colonnes et de lignes des images chiffrées $A(i,j)$ et $B(i,j)$. Les valeurs idéales de ces paramètres sont données par :

$NPCR = 99.61$ et $UACI = 33.46$.

Un NPCR ($NPCR > 99.5810$) et un UACI ($33.3445 \leq UACI \leq 33.5826$) [] élevés impliquent une grande résistance du chiffrement aux attaques différentielles.

Les résultats de calcul des paramètres statistiques (UACI et NPCR) des images chiffrées issues de l'algorithme de chiffrement sur 4 bits sont groupés dans le tableau ci-dessous.

TABLE 3.3 – Résultats des tests de l'UACI et NPCR

Images	Couleurs	UACI (256 bits)	NPCR (256 bits)	UACI (2048 bits)	NPCR (2048 bits)
Lena	Rouge	33.4238	99.6078	33.4431	99.6239
	Verte	33.4896	99.6098	33.4277	99.5922
	Bleue	33.5038	99.6136	33.5368	99.6296
Baboon	Rouge	33.4987	99.6048	33.4614	99.6002
	Verte	33.4434	99.5956	33.5194	99.6178
	Bleue	33.4855	99.6243	33.4075	99.6094
Airplane	Rouge	33.4631	99.6063	33.4199	99.5899
	Verte	33.4720	99.6208	33.4583	99.6155
	Bleue	33.5359	99.5884	33.4986	99.6162
Peppers	Rouge	33.5103	99.6117	33.5019	99.6014
	Verte	33.3958	99.6426	33.5445	99.5899
	Bleue	33.3403	99.6315	33.4066	99.6273

Ces résultats ont été obtenus en effectuant 3 tours de l'algorithme de chiffrement. Les valeurs obtenus sur le tableau sont proches des valeurs idéales, ce qui prouve que notre algorithme proposé est efficace. Les images utilisés pour effectuer ces tests sont les images couleurs standard de taille 512×512 utilisés habituellement en cryptographie d'images.

3.2.2) Analyse de l'espace clef

L'espace de clef représente l'ensemble des combinaisons possibles de clefs nécessaire pour effectuer les opérations de chiffrement et de déchiffrement. La taille minimale d'une clef de chiffrement doit être de 256 bits et elle est largement suffisante pour résister contre les différentes attaques à force brute.

L'un des principaux avantages du chiffrement proposé est l'extensibilité de son espace de clef. Les ordinateurs quantiques menacent les ordinateurs classiques car ils peuvent forcer, dans un laps de temps relativement court, les normes actuelles de cryptage à clé publique. Ces ordinateurs peuvent briser des algorithmes cryptographiques standards et complexes tels que RSA et ECC. C'est pourquoi, plusieurs schémas de cryptographie post-quantique (PQC) sont développés pour pouvoir résister à la fois aux attaques classiques et quantiques.

Ainsi, pour résister aux menaces quantiques, on se propose dans ce travail de thèse, d'étendre l'espace de la clef de chiffrement à une taille de 256 bits.

En effet, la plupart des chiffrements basés sur le chaos présente un grand espace de clef dont on ne peut pas justifier facilement qu'elles sont différentes. Cependant, la sensibilité de la clé est vérifiée en changeant son bit le moins significatif (LSB). Cependant, grâce à notre approche, l'espace de clef peut être étendu à volonté sans toutefois détruire l'indépendance des clefs.

Cette approche consiste, à affecter à chaque sous-image j, une sous clef $(\mathbf{a}(k), \mathbf{b}(k), \mathbf{c}(k), \mathbf{d}(k), \mathbf{e}(k), \mathbf{f}(k), \mathbf{g}(k), \mathbf{h}(k))$ correspondant à la récurrence $QACM_k$, avec $k = (1 + j) \bmod \kappa$.

Chaque sous image j est chiffrée avec une récurrence $QACM_k$, de sorte que, toute permutation dans l'ensemble $QACM_{k\,1\leq k\leq \kappa}$ influe sur le comportement du cryptogramme, donnant ainsi la possibilité d'étendre l'espace de clef.

3.2.3) Sensibilité de la clef

La sensibilité de la clef mesure la sensibilité de l'algorithme de chiffrement suite à une légère modification de celle-ci. Une sensibilité élevée de la clef est nécessaire pour prévenir contre les différentes attaques de l'image en clair et la cryptanalyse.

Pour évaluer la sensibilité de notre clef de chiffrement, nous avons chiffré la même image en

utilisant 2 clefs légèrement différentes . Les clefs sont données par :
$K1 = azertyuiopqsdfgjazertyuiopqsdfg0$ et $K2 = azertyuiopqsdfgjazertyuiopqsdfg1$.
Comme nous pouvons le remarquer, les 2 clefs diffèrent l'un de l'autre à travers un caractère.
. Nous avons répété la même expérience avec une clé de 2048 bits. Nous avons défini la clé

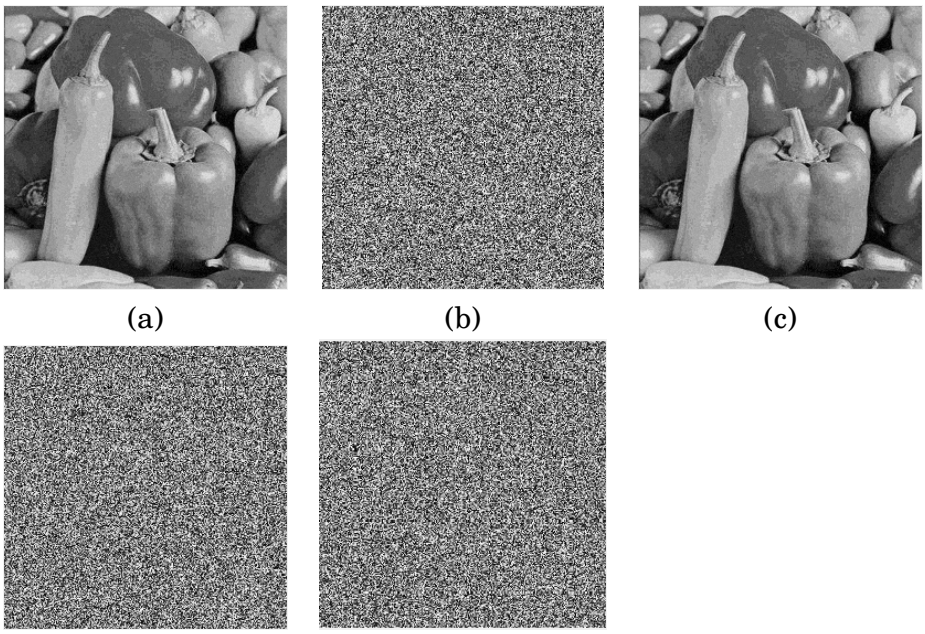


FIGURE 3.3 – Résultats test de simulation sur 4 bits : a) Pepper 256×256 ; b)Chiffrement avec la clef K1 ; c) Déchiffrement avec la clef K1 ; (d),(e) Déchiffrement avec K2

de 2048 bits comme $K1 = K1K1K1K1K1K1K1K1$ et une clé K2 différente d'un caractère comme $K2 = K1K1K1K1K1K1K1K2$. La sensibilité de la clef a été évaluée dans les mêmes conditions que pour celle de 256 bits et les résultats sont présentés dans le tableau 3.3. Il montre que la sensibilité de l'analyse de la clé est la même que dans le cas de 256 bits. Par conséquent, nous pouvons conclure que le chiffrement proposé présente un espace de clef extensible qui peut être modifié.

3.2.4) Analyse statistique

3.2.4.1) *Test aléatoire*

Le test aléatoire d'une séquence dynamique, est l'un des critères pour évaluer les performances du niveau de sécurité d'un algorithme de chiffrement. Bien que différents tests d'aléa furent proposés jusqu'à présent, il n'existe aucune séquence dynamique qui réussit tous les tests d'aléa. L'un des tests d'aléa les plus populaire jusqu'à nos jours est le SP800-22 [99]. En effet,le SP800-22 fournit une série de test statistiques (STS) composée de 15 tests.

Comme la récurrence du chat d'Arnold discret (QACM), la 4D-PWLCM est également chaotique lorsqu'il est utilisé dans un espace de phase continu, c'est à dire $N = 1$. Lorsque celle ci est dans un espace de phase discret, l'évaluation du calcul des périodes présenté au chapitre 2, a démontré qu'il conservait les mêmes propriétés que celle d'un système chaotique.

Pour prouver l'efficacité de la 4D-PWLCM, nous avons brouillé périodiquement les pixels d'une image de taille $2^{13} \times 2^{13}$. Pour effectuer le test statistique NIST800-22, les paramètres de contrôles et conditions initiales ont été prises tels que :

$q_0 = 1, r_0 = 2; x_0, y_0 \in [0, 2^{13} - 1], a_1 = 1, b_1 = 2, e_1 = 0, f_1 = 1, c_1 = 3, d_1 = 3, g_1 = 3, h_1 = 3$;

La période correspondante à ces paramètres est de 148740480. Le test statistique NIST800-22 a été effectué après 30 itérations de brouillage d'image avec la 4-D PWLCM d'une part et la QACM d'autre part. L'ensemble de données a été divisé en 100 ensembles de 1000000 bits et les résultats obtenus sont résumés dans le tableau ci-dessous.

TABLE 3.4 – Résultats du test de Nist

	4D-QACM		4D-PWLCM	
tests statistiques	P-value	Proportion	P-value	Proportion
1.Frequency	0.0	22/100	0.319084	99/100
2.Block frequency	0.0	0/100	0.574903	98/100
3.Cumulative sums (forward)	0.0	0/100	0.289667	100/100
4.Runs	0.0	0/100	0.657933	99/100
5.Longest run	0.0	0/100	0.017912	100/100
6.Rank	0.0	0/100	0.657933	100/100
7.FFT	0.0	0/100	0.319084	98/100
8.Non-overlapping template	0.0	100/100	0.971699	100/100
9.Overlapping template	0.0	0/100	0.574903	99/100
10.Universal	0.0	0/100	0.066882	100/100
11.Approximate entropy	0.0	0/100	0.304126	100/100
12.Random excursions	0.0	0/61	0.957319	61/61
13.Random excursions variant	0.0	0/61	0.957319	61/61
14.Serial	0.0	0/100	0.955835	98/100
15.Linear complexity	0.0	0/100	0.955835	100/100

3.2.5) Analyse des performances de rapidité

La vitesse d'exécution de l'algorithme proposé sur 4 bits est évaluée en utilisant le logiciel Matlab 2018b et une machine ayant les caractéristiques comme décrites au paragraphe précédent. Les images utilisées pour effectuer ces tests sont les images en niveau de gris du Cameraman et de Lena respectivement de tailles 256×256 et 512×512. Nous avons comparé nos résultats avec ceux présentés dans les travaux [100] et [101]. Nous observons que, le temps d'exécution de l'algorithme proposé sur 4 bits reste plus faible que celui présenté dans [17]. En effet, pour chiffrer une image de taille 512×512 par exemple, le temps d'exécution est en moyenne de $0.4478s$ obtenu juste après 2 itérations (2 rounds). Ce faible temps de calcul vient du fait que, les conversions de données impliquées dans l'algorithme n'interviennent pas. Par exemple, la décomposition d'une pixel de l'image en niveau de gris en deux entiers codés sur 4 bits $q(t)$ et $r(t)$ implique une division et une opération de modulation. Ainsi, l'algorithme proposé permet de gagner en temps de calcul et également adapté à une implémentation matérielle à faible cout. La comparaison faite avec la référence [17] montre que l'algorithme proposé s'exécute moins rapidement que celle ci. Néanmoins, une implémentation matérielle dans celle ci [17] nécessite une arithmétique à virgule flottante qui est plus coûteuse.
Le tableau regroupant donc les comparaisons des temps d'exécution des algorithmes de chiffrement de certains travaux avec le nôtre est donnée ci-dessous.

TABLE 3.5 – Résultats des temps de simulations

Images	temps d'exécution(s)			
	cryptosystème sur 4 bits	Ref [17]	Ref [100]	Ref [101]
Cameraman 256×256	0.1335	0.0202	0.1789	0.1950
Lena 512×512	0.4478	0.0708	0.6639	0.6500

Il est à noter que, les durées de simulations obtenues dans ce tableau sont faites après 2 tours de la 4D-PWLCM combiné à la 8D-PWLCM tel que présenté à l'organigramme donné précédemment.
Les résultats présentés dans ce tableau montre que l'algorithme proposé sur 4 bits est rapide en temps de calcul et peut être utile pour une application à temps réel. De plus, son implémentation matérielle nécessite moins de composants électroniques comparativement à

d'autres cryptosystèmes présentés dans les références précitées dans le tableau ci-dessous qui nécessitent un cout de matériel conséquent.

3.3) Présentation de l'algorithme sur 2 bits

Les différentes étapes sur l'algorithme de chiffrement sur 4 bits restent valables sur 2 bits. Dans le cas sur 2 bits, la différence s'effectue juste au niveau de l'étape de diffusion des pixels. En effet, sur 2 bits, l'opération de diffusion s'effectue en utilisant la récurrence 4D-PWLCM proposé tandis que sur 4 bits, les opérations de confusion et de diffusion s'effectuait simultanément en utilisant la 4D-PWLCM. Ainsi, l'organigramme présentant l'algorithme de chiffrement sur 2 bits peut être schématisé ci-dessous.

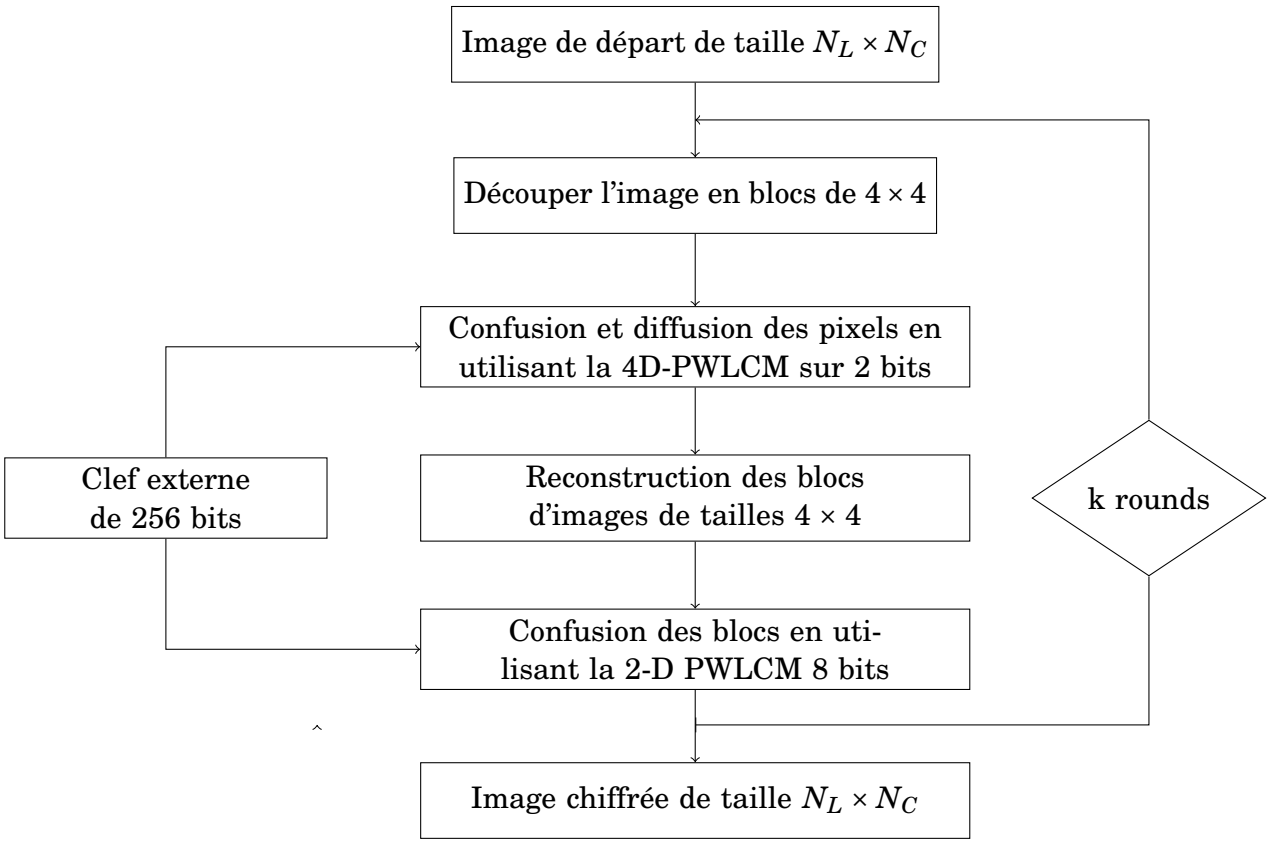


FIGURE 3.4 – Organigrame de chiffrement utilisant la 4-D PWLCM sur 2 bits

Comme nous pouvons le remarquer, le processus de chiffrement de la récurrence d'Arnold en dimension 4 modifié en utilisant une précision de 2 bits nécessite moins d'opérations comparativement à celle sur 4 bits. En effet, sur 2 bits, 2 opérations ont été utilisées : la diffusion d'une part et la permutation d'autre part.

3.4) Résultats et analyse de sécurité sur 2 bits

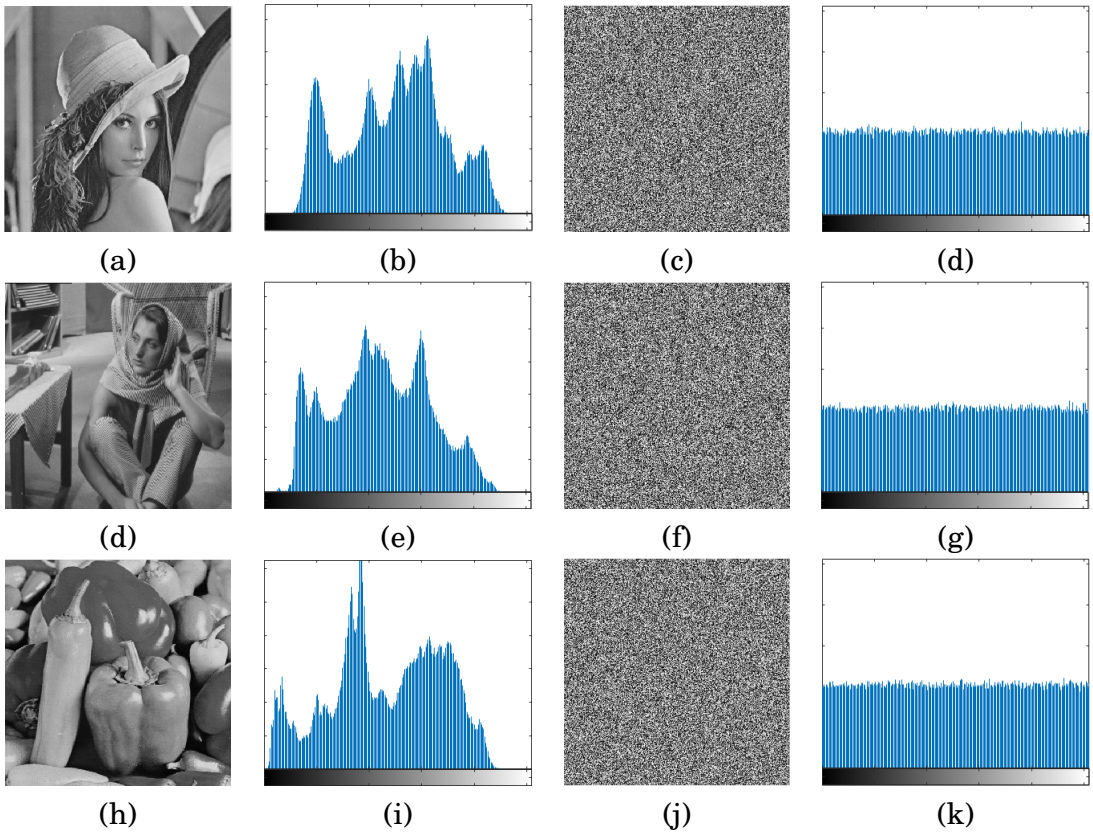


FIGURE 3.5 – Quelques tests sur les images de tailles 512×512 : a) lena, d) Barbara et h) Peppers

.

3.4.1) Principe de repartition de la clef externe

Le principe de distribution des valeurs issues de la clef externe sur 2 bits s'effectue comme dans le cas étudié sur 4 bits. Dans ce cas, pour avoir une taille de clef de sécurité minimum de 256 bits, il suffit d'augmenter la taille des paramètres de contrôles de la récurrence d'Arnold 4D sur 2 bits qui permet d'effectuer simultanément les opérations de confusion et de diffusion des pixels de l'image. De plus pour guarantir la robustesse du cryptosystème et l'extensibilité de la clef externe, une opération de permutation supplémentaire est effectuée en utilisant la récurrence d'Arnold 4D. Ainsi, les systèmes 4D sur 2 bits et 2D s'écriront comme suit :

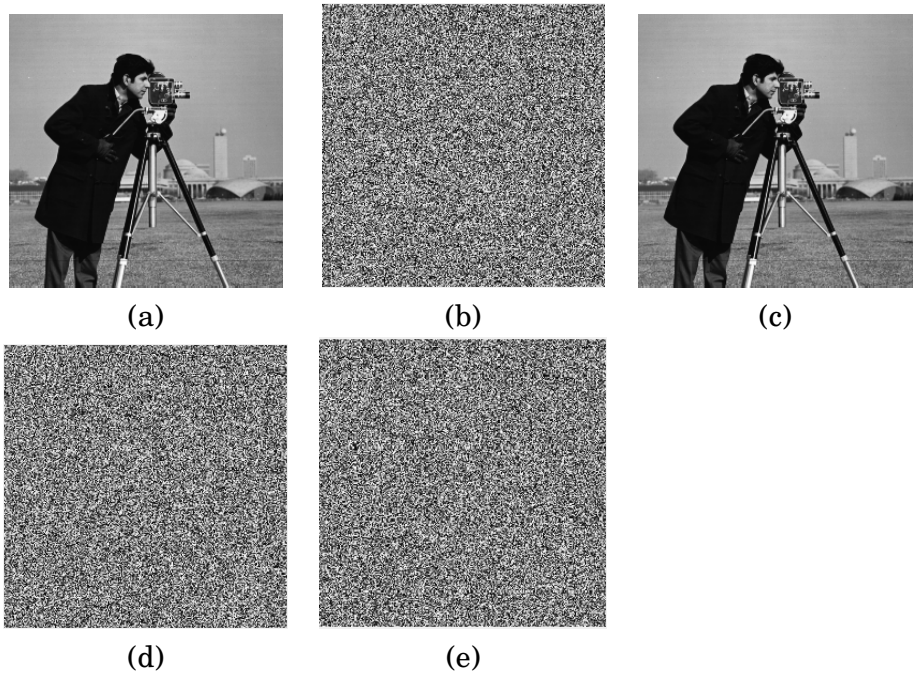
$$\begin{cases} x(t+1) = x(t) + \alpha y(t) + (\mathbf{a}(k) + y(t)) \mod \mathbf{c}(k) \\ y(t+1) = y(t) + \beta x(t+1) + (\mathbf{b}(k) + x(t+1)) \mod \mathbf{d}(k) \\ q(t+1) = q(t) + \gamma y(t+1) + (\mathbf{e}(k) + y(t+1)) \mod \mathbf{g}(k) \\ r(t+1) = r(t) + \zeta q(t+1) + (\mathbf{f}(k) + q(t+1)) \mod \mathbf{h}(k) \end{cases} \mod 4 \tag{3.19}$$

où, $k = 1 + j \ mod \kappa$.

j étant le sous bloc image de taille 4×4 et κ correspond à la taille du vecteur de paramètre de contrôles. Dans ce cas de 2 bits, la taille minimale est $\kappa = 16$ (pour avoir une taille de clef de 256 bits).

$$\begin{cases} x(t+1) = (x(t) + \alpha y(t) + \sum_{k=1}^{16} (\mathbf{a_1}(k) + y(t)) \mod \mathbf{c_1}(k)) \mod m_1 \\ y(t+1) = (\beta x(t+1) + y(t) + \sum_{k=1}^{16} (\mathbf{b_1}(k) + x(t+1)) \mod \mathbf{d_1}(k)) \mod m_2 \end{cases} \tag{3.20}$$

Les paramètres de contrôles a_1, b_1, c_1 et d_1 sont définit comme au paragraphe précédent.



(a) (b) (c)

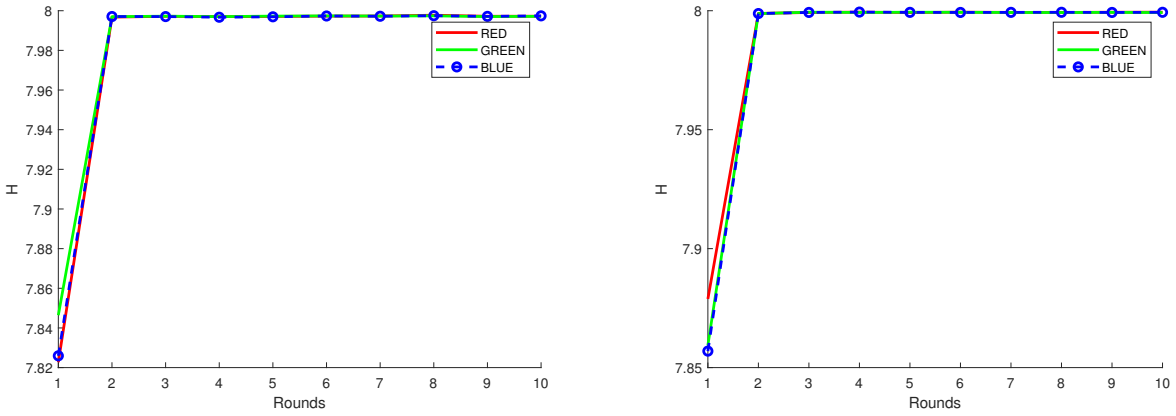
(d) (e)

FIGURE 3.6 – Résultats test de simulation sur 2 bits : a) Cameraman 256×256 ; b) Chiffrement avec la clef K1 ; c) Déchiffrement avec la clef K1 ; (d),(e) Dechiffrement avec K2

.

3.4.2) *Analyse des performances*

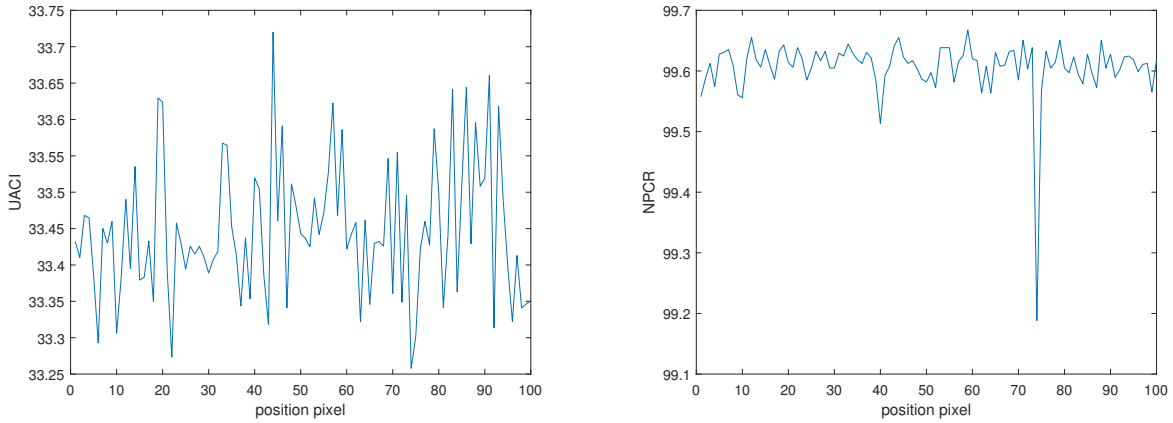
Comme nous l'avons définit précédemment, pour évaluer les performances d'un algorithme de chiffrement, plusieurs métriques sont utilisées telles que : le calcul de l'UACI, du NPCR, de l'entropie, les coefficients de corrélations, etc. Ainsi, nous pouvons représenter ci-dessous, la variation de l'entropie de permutation effectuée sur le cryptosystème basé sur la récurrence d'Arnold 4D sur 2 bits en utilisant comme entrées les images de tailles 256×256 de Lena et 512×512 de Barbara.



(a) (b)

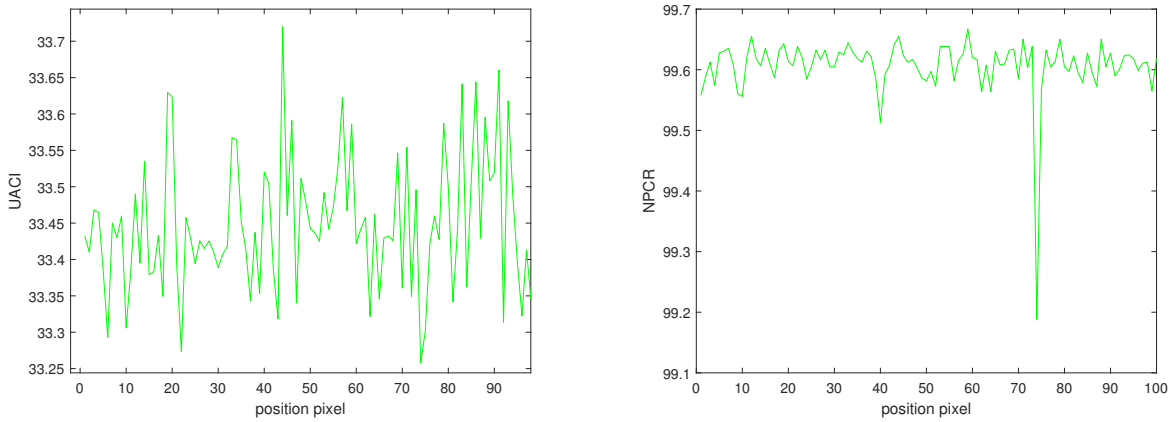
FIGURE 3.7 – Représentation de l'entropie H sur 2 bits des images chiffrées couleur de : a) Lena 256×256) ; b) Barbara 512×512

Après analyse de la courbe, il convient de remarquer que, l'entropie de permutation tend la valeur idéale qui est de 8 à partir de 3 rounds effectués sur le cryptosystème.



(a) (b)

FIGURE 3.8 – Représentation de l'UACI et NPCR sur 2 bits des images chiffrées couleur bleue de Lena 256×256 de : a) UACI) ; b) NPCR



(a) (b)

FIGURE 3.9 – Représentation de l'UACI et NPCR sur 2 bits des images chiffrées couleur Verte de Lena 256×256 de : a) UACI) ; b) NPCR

D'après analyse des courbes ci- dessous, on observe que les valeurs moyennes de calculs de l'UACI et le NPCR sont proches des valeurs idéales connues dans la littérature (UACI= $99,61$ et NPCR =$33 : 46$) ; Ce qui prouve que le cryptosystème proposé sur 2 bits est également adapté pour des applications cryptographiques.

TABLE 3.6 – Résultats des temps de simulations

Images	temps d'exécution(s)	
	cryptosystème sur 4 bits	cryptosystème sur 2 bits
Cameraman 256×256	0.1335	0.1225
Lena 512×512	0.4478	0.3941

Conclusion

Au terme de ce chapitre, il était question de présenter les résultats relatives à l'algorithme de chiffrement sur 4 bits puis sur 2 bits de l'algorithme de chiffrement utilisant la récurrence d'Arnold. Les résultats ont prouvé, que les 2 algorithmes proposés sont efficaces et robustes contre les différentes attaques informatiques. les histogrammes obtenus des images chiffrées, ont prouvé leur efficacité à travers la répartition uniforme de la distribution des pixels. Ce qui montre une fois de plus que les images chiffrées étaient efficaces sur les différentes précisions. Il convient aussi de remarquer que l'espace de distribution des clefs est extensible comparativement à d'autres types d'algorithmes de chiffrement.

CONCLUSION GÉNÉRALE ET PERSPECTIVES

Dans ce travail, nous avons implémenté la RALM en dimension 4 codée respectivement sur les précisions de 2 et 4 bits en s'inspirant de la récurrence du chat d'Arnold. L'intérêt d'un tel travail était de renforcer le niveau de sécurité des cryptosystèmes chaotiques, en plus de réduire de moitié le temps imparti aux opérations de confusion et diffusion en les combinant en une seule opération. Cette combinaison des opérations a été rendue possible en effectuant une décomposition linéaire de chaque pixel en fonction des différentes conditions initiales prises par le système dans l'espace de phase. La réduction de la précision quant à elle a permis de réduire la complexité du cryptosystème utilisant la récurrence, donc de réduire le coût matériel lors de son implémentation. Après investigation, nous avons vérifié que la période de RALM est largement supérieure à celle de la récurrence d'Arnold classique. Un tel résultat permet effectivement d'améliorer la sécurité des cryptosystèmes qui en dépendent.

Nous avons dans un premier temps développé la RALM 4D en ajoutant des termes non linéaires sur chaque variable d'état de la récurrence du chat d'Arnold classique et obtenu après simulation, une période de l'ordre de 10^8 pour une précision de 4 bits. Le système ainsi obtenu présente plusieurs paramètres de contrôles, ce qui permet de concevoir des cryptosystèmes avec un grand espace de clefs garantissant ainsi la sécurité de ces cryptosystèmes. Tenant compte des performances de la RALM 4D, nous avons conçu un cryptosystème à espace de clef extensible. De plus, ledit cryptosystème combine les opérations de confusion et de diffusion en une seule, ce qui contribue à augmenter sa rapidité. L'analyse statistique des résultats a montré que le cryptosystème proposé est robuste contre les attaques connues. Cependant une

grande précision implique également une forte consommation d'énergie. Pour davantage réduire le coût matériel et la consommation d'énergie, nous avons développé un cryptosystème intégrant la RALM 4D codée sur 2 bits. Bien que la période ne soit plus que 360, l'analyse statistique du cryptosystème montre une meilleure robustesse contre les attaques à forces brutes connues. Une telle performance est liée à son architecture qui utilise moins de temps de calcul que celui basé sur la RALM codée sur 4 bits, pour compenser la faible période des orbites de la RALM codée sur 2 bits.

En perspectives, nous envisageons dans un futur proche, réduire davantage la précision, tout en augmentant la période des trajectoires de la RALM. Un tel travail permettrait de simplifier davantage l'architecture des cryptosystèmes sans toutefois altérer le niveau de sécurité. De plus, vu son architecture algorithmique, il serait possible également d'envisager une implémentation hardware de ce cryptosystème soit en utilisant des circuits électroniques de bases (résistors, additionneurs, multiplexeurs, etc.) où les cartes et circuits électroniques programmables (FPGA, Microcontrolleur, etc.). De plus, une telle simplification de l'architecture algorithmique permettrait d'optimiser à la fois le coût matériel et la consommation d'énergie, garantissant ainsi une meilleure performance de nos algorithmes dans les systèmes embarqués.

BIBLIOGRAPHIE

[1] Alan Wolf, Jack B Swift, Harry L Swinney, and John A Vastano. Chaotic behavior in the hénon mapping. *Communications in Mathematical Physics*, 1979.

[2] Kathleen T Alligood, Tim D Sauer, James A Yorke, and David Chillingworth. Chaos : an introduction to dynamical systems. *SIAM Review*, 1998.

[3] AJ Menezes, PC van Oorschot, and SA Vanstone. Mathematical background. *Handbook of Applied Cryptography, CRC Press, Boca Raton, Florida, USA*, 1997.

[4] Hoonjae Lee and Sangjae Moon. Parallel stream cipher for secure high-speed communications. *Signal Processing*, 2002.

[5] Wei Li, Des McLernon, Jing Lei, Mounir Ghogho, Syed Ali Raza Zaidi, and Huaihai Hui. Mathematical model and framework of physical layer encryption for wireless communications. In *2018 IEEE Globecom Workshops (GC Wkshps)*.

[6] Cristian Chiţu and Manfred Glesner. An fpga implementation of the aes-rijndael in ocb/ecb modes of operation. *Microelectronics Journal*, 2005.

[7] Yang Xiao, Hsiao-Hwa Chen, Xiaojiang Du, and Mohsen Guizani. Stream-based cipher feedback mode in wireless error channel. *IEEE Transactions on Wireless Communications*, 2009.

[8] Man Young Rhee. *Internet security : cryptographic principles, algorithms and protocols*. John Wiley & Sons, 2003.

[9] Zhi-liang Zhu, Wei Zhang, Kwok-wo Wong, and Hai Yu. A chaos-based symmetric image encryption scheme using a bit-level permutation. *Information Sciences*, 2011.

[10] Guodong Ye and Kwok-Wo Wong. An efficient chaotic image encryption algorithm based on a generalized arnold map. *Nonlinear dynamics*, 2012.

[11] Guanrong Chen, Yaobin Mao, and Charles K Chui. A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos, Solitons & Fractals*, 2004.

[12] Yue Wu, Gelan Yang, Huixia Jin, and Joseph P Noonan. Image encryption using the two-dimensional logistic chaotic map. *Journal of Electronic Imaging*, 2012.

[13] Ping Ping, Feng Xu, Yingchi Mao, and Zhijian Wang. Designing permutation–substitution image encryption networks with henon map. *Neurocomputing*, 2018.

[14] Yaobin Mao, Guanrong Chen, and Shiguo Lian. A novel fast image encryption scheme based on 3d chaotic baker maps. *International Journal of Bifurcation and chaos*, 2004.

[15] Vladimir Igorevich Arnol'd. *Mathematical methods of classical mechanics*. Springer Science & Business Media, 2013.

[16] Zhi-Hong Guan, Fangjun Huang, and Wenjie Guan. Chaos-based image encryption algorithm. *Physics letters A*, 2005.

[17] JS Armand Eyebe Fouda, J Yves Effa, Samrat L Sabat, and Maaruf Ali. A fast chaotic block cipher for image encryption. *Communications in Nonlinear Science and Numerical Simulation*, 2014.

[18] Freeman J Dyson and Harold Falk. Period of a discrete cat mapping. *The American Mathematical Monthly*, 1992.

[19] Gonzalo Alvarez and Shujun Li. Some basic cryptographic requirements for chaos-based cryptosystems. *International journal of bifurcation and chaos*, 2006.

[20] Narendra K Pareek, Vinod Patidar, and Krishan K Sud. Image encryption using chaotic logistic map. *Image and vision computing*, 2006.

[21] N Bourbakis and Christos Alexopoulos. Picture data encryption using scan patterns. *Pattern Recognition*, 1992.

[22] Philippe Refregier and Bahram Javidi. Optical image encryption based on input plane and fourier plane random encoding. *Optics letters*, 1992.

[23] Josef Scharinger. Fast encryption of image data using chaotic kolmogorov flows. *Journal of Electronic imaging*, 1998.

[24] A ALI-PACHA, N Hadj-Said, A M'HAMED, and A Belghoraf. Chaos crypto système base sur l'attracteur de clifford. In *SETIT 5th International Conference : Sciences of Electronic, Technologies of Information and Telecommunications*.

[25] Kathleen T Alligood, Tim D Sauer, James A Yorke, and David Chillingworth. Chaos : an introduction to dynamical systems. *SIAM Review*, 1998.

[26] Shahram Etemadi Borujeni and Mohammad Eshghi. Chaotic image encryption design using tompkins-paige algorithm. *Mathematical Problems in Engineering*, 2009.

[27] Gregory L Baker and Jerry P Gollub. *Chaotic dynamics : an introduction*. Cambridge university press, 1996.

[28] Shamsa Kanwal, Saba Inam, Mohamed Tahar Ben Othman, Ayesha Waqar, Muhammad Ibrahim, Fariha Nawaz, Zainab Nawaz, and Habib Hamam. An effective color image encryption based on henon map, tent chaotic map, and orthogonal matrices. *Sensors*, 2022.

[29] Ali Kanso. Self-shrinking chaotic stream ciphers. *Communications in nonlinear science and numerical simulation*, 2011.

[30] Kathleen T Alligood, Tim D Sauer, James A Yorke, and David Chillingworth. Chaos : an introduction to dynamical systems. *SIAM Review*, 1998.

[31] Ali Mansouri and Xingyuan Wang. A novel one-dimensional sine powered chaotic map and its application in a new image encryption scheme. *Information Sciences*, 2020.

[32] Der-Chyuan Lou and Chia-Hung Sung. "A steganographic scheme for secure communications based on the chaos and Euler theorem," ieee trans. multimedia, vol. 6, no. 3, pp. 501–509, jun. 2004. *IEEE Transactions on Multimedia*, 2004.

[33] Guanrong Chen, Yaobin Mao, and Charles K Chui. A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos, Solitons & Fractals*, 2004.

[34] Chittaranjan Pradhan, Vilakshan Saxena, and Ajay Kumar Bisoi. Imperceptible watermarking technique using arnold's transform and cross chaos map in dct domain. *International Journal of Computer Applications*, 2012.

[35] Christos Volos, Ioannis Kyprianidis, Ioannis Stouboulos, and Sundarapandian Vaidyanathan. Random bit generator based on non-autonomous chaotic systems. In *Handbook of Research on Advanced Intelligent Control Engineering and Automation*.

[36] ZH Liu and WQ Zhu. Homoclinic bifurcation and chaos in simple pendulum under bounded noise excitation. *Chaos, Solitons & Fractals*, 2004.

[37] Edward N Lorenz. Detemainistic mmperkxtic flow. *Journal of the Atnxherie Sciences*, 1963.

[38] Jing Hu, Wen-wen Tung, Jianbo Gao, and Yinhe Cao. Reliability of the 0-1 test for chaos. *Physical Review E*, 2005.

[39] Robert Shaw. Strange attractors, chaotic behavior, and information flow. *Zeitschrift für Naturforschung A*, 1981.

[40] H Haken. At least one lyapunov exponent vanishes if the trajectory of an attractor does not contain a fixed point. *Physics Letters A*, 1983.

[41] Alan Wolf, Jack B Swift, Harry L Swinney, and John A Vastano. Determining lyapunov exponents from a time series. *Physica D : nonlinear phenomena*, 1985.

[42] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 1948.

[43] Holger Kantz and Thomas Schreiber. *Nonlinear time series analysis*. Cambridge university press, 2004.

[44] Steven M Pincus. Approximate entropy as a measure of system complexity. *Proceedings of the National Academy of Sciences*, 1991.

[45] Peter Grassberger and Itamar Procaccia. Measuring the strangeness of strange attractors. *The theory of chaotic attractors*, 2004.

[46] Stefano Galatolo. Orbit complexity and data compression. *Discrete and Continuous Dynamical Systems*, 2001.

[47] Norbert Marwan, M Carmen Romano, Marco Thiel, and Jürgen Kurths. Recurrence plots for the analysis of complex systems. *Physics reports*, 2007.

[48] Matthäus Staniek and Klaus Lehnertz. Symbolic transfer entropy. *Physical review letters*, 2008.

[49] José Amigó. *Permutation complexity in dynamical systems : ordinal patterns, permutation entropy and all that*. Springer Science & Business Media, 2010.

[50] Christoph Bandt and Bernd Pompe. Symbolic transfer entropy. *Physical review letters*, 2002.

[51] John Von Neumann. Various techniques used in connection with random digits. *John von Neumann, Collected Works*, 1963.

[52] John Kelsey, Bruce Schneier, David Wagner, and Chris Hall. Cryptanalytic attacks on pseudorandom number generators. In *International workshop on fast software encryption*.

[53] Ta Thi Kim Hue and Thang Manh Hoang. Complexity and properties of a multidimensional cat-hadamard map for pseudo random number generation. *The European Physical Journal Special Topics*, 2017.

[54] Shijian Cang, Zhijun Kang, and Zenghui Wang. Pseudo-random number generator based on a generalized conservative sprott-a system. *Nonlinear Dynamics*, 2021.

[55] Jianwen Lv, Xiaodong Li, Tao Yang, Haoyang Yu, and Beisheng Liu. A general pseudorandom number generator based on chaos. In *4th EAI International conference on robotic sensor networks*.

[56] Borislav Stoyanov and Tsvetelina Ivanova. Chaosa : Chaotic map based random number generator on arduino platform. In *AIP Conference Proceedings*.

[57] Fei Yu, Lixiang Li, Binyong He, Li Liu, Shuai Qian, Zinan Zhang, Hui Shen, Shuo Cai, and Yi Li. Pseudorandom number generator based on a 5d hyperchaotic four-wing memristive system and its fpga implementation. *The European Physical Journal Special Topics*, 2021.

[58] Erdinc Avaroğlu. Pseudorandom number generator based on arnold cat map and statistical analysis. *Turkish Journal of Electrical Engineering and Computer Sciences*, 2017.

[59] Djeugoue Hermann, Gnyamsi Gaetan Gildas, Jean Sire Armand Eyebe Fouda, and Wolfram Koepf. On the implementation of large period piece-wise linear arnold cat map. *Multimedia Tools and Applications*, 2022.

[60] Harry Nyquist. Thermal agitation of electric charge in conductors. *Physical review*, 1928.

[61] Mario Stipčević. Fast nondeterministic random bit generator based on weakly correlated physical events. *Review of scientific instruments*, 2004.

[62] Mario Stipčević and Çetin Kaya Koç. True random number generators. In *Open Problems in Mathematics and Computational Science*.

[63] Chris Sundt. Information security and the law. *Information Security Technical Report*, 2006.

[64] William Stallings, Lawrie Brown, Michael D Bauer, and Michael Howard. *Computer security : principles and practice*. Pearson Upper Saddle River, 2012.

[65] HX Mel, Doris M Baker, and Steve Burnett. *Cryptography decrypted*. Addison-Wesley Reading, MA, 2001.

[66] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 1978.

[67] Michael O Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Massachusetts Inst of Tech Cambridge Lab for Computer Science, 1979.

[68] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 1985.

[69] FIPS Pub. Data encryption standard (des). *FIPS PUB*, 1999.

[70] Xuejia Lai and James L Massey. A proposal for a new block encryption standard. In *Workshop on the Theory and Application of of Cryptographic Techniques*.

[71] Morris J Dworkin, Elaine B Barker, James R Nechvatal, James Foti, Lawrence E Bassham, E Roback, James F Dray Jr, et al. Advanced encryption standard (aes). *FIPS PUB*, 2001.

[72] Bruce Schneier. Description of a new variable-length key, 64-bit block cipher (blowfish). In *International Workshop on Fast Software Encryption*.

[73] RC-W Phan and Mohammad Umar Siddiqi. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on computers*, 2006.

[74] AJ Menezes, PC van Oorschot, and SA Vanstone. A public key cryptosystem and a signature scheme based on discrete logarithms. *Handbook of Applied Cryptography, CRC Press, Boca Raton, Florida, USA*, 1997.

[75] Mark Stamp and Richard M Low. *Applied cryptanalysis : breaking ciphers in the real world*. John Wiley & Sons, 2007.

[76] AJ Menezes, PC van Oorschot, and SA Vanstone. Mathematical background. *Handbook of Applied Cryptography, CRC Press, Boca Raton, Florida, USA*, 1997.

[77] Rhouma Rhouma and Safya Belghith. Cryptanalysis of a spatiotemporal chaotic image/-video cryptosystem. *Physics Letters A*, 2008.

[78] AJ Menezes, PC van Oorschot, and SA Vanstone. Mathematical background. *Handbook of Applied Cryptography, CRC Press, Boca Raton, Florida, USA*, 1997.

[79] William Stallings. Nist block cipher modes of operation for confidentiality. *Cryptologia*, 2010.

[80] Zbigniew Kotulski and Janusz Szczepański. Discrete chaotic cryptography. *Annalen der Physik*, 1997.

[81] Howard M Heys. Analysis of the statistical cipher feedback mode of block ciphers. *IEEE Transactions on Computers*, 2003.

[82] Xiaoling Huang and Guodong Ye. An efficient self-adaptive model for chaotic image encryption algorithm. *Communications in Nonlinear Science and Numerical Simulation*, 2014.

[83] Hongjun Liu and Xingyuan Wang. Color image encryption using spatial bit-level permutation and high-dimension chaotic system. *Optics Communications*, 2011.

[84] Yue Wu, Joseph P Noonan, Gelan Yang, and Huixia Jin. Image encryption using the two-dimensional logistic chaotic map. *Journal of Electronic Imaging*, 2012.

[85] Shiguo Lian, Jinsheng Sun, and Zhiquan Wang. A block cipher based on a suitable use of the chaotic standard map. *Chaos, Solitons & Fractals*, 2005.

[86] Ping Ping, Feng Xu, Yingchi Mao, and Zhijian Wang. Designing permutation–substitution image encryption networks with henon map. *Neurocomputing*, 2018.

[87] Yaobin Mao, Guanrong Chen, and Shiguo Lian. A novel fast image encryption scheme based on 3d chaotic baker maps. *International Journal of Bifurcation and chaos*, 2004.

[88] Benyamin Norouzi, Seyed Mohammad Seyedzadeh, Sattar Mirzakuchaki, and Mohammad Reza Mosavi. A novel image encryption based on row-column, masking and main diffusion processes with hyper chaos. *Multimedia Tools and Applications*, 2015.

[89] Guanrong Chen, Yaobin Mao, and Charles K Chui. A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos, Solitons & Fractals*, 2004.

[90] Ping Ping, Jinyang Fan, Yingchi Mao, Feng Xu, and Jerry Gao. A chaos based image encryption scheme using digit-level permutation and block diffusion. *IEEE Access*, 2018.

[91] Wei Zhang, Hai Yu, Yu-li Zhao, and Zhi-liang Zhu. Image encryption based on three-dimensional bit matrix permutation. *Signal Processing*, 2016.

[92] Weijia Cao, Yicong Zhou, CL Philip Chen, and Liming Xia. Medical image encryption using edge maps. *Signal Processing*, 2017.

[93] Guodong Ye and Xiaoling Huang. An efficient symmetric image encryption algorithm based on an intertwining logistic map. *Neurocomputing*, 2017.

[94] Sha-Sha Yu, Nan-Run Zhou, Li-Hua Gong, and Zhe Nie. An efficient symmetric image encryption algorithm based on an intertwining logistic map. *Optics and Lasers in Engineering*, 2020.

[95] Vladimir Igorevich Arnold and André Avez. *Ergodic problems of classical mechanics*. Benjamin, 1968.

[96] Freeman J Dyson and Harold Falk. Period of a discrete cat mapping. *The American Mathematical Monthly*, 1992.

[97] Ruisong Ye and Haiying Zhao. An efficient chaos-based image encryption scheme using affine modular maps. *International Journal of Computer Network and Information Security*, 2012.

[98] Anak Agung Putri Ratna, Frenzel Timothy Surya, Diyanatul Husna, I Ketut Eddy Purnama, Ingrid Nurtanio, Afif Nurul Hidayati, Mauridhi Hery Purnomo, Supeno Mardi Susiki Nugroho, and Reza Fuad Rachmadi. Chaos-based image encryption using arnold's cat map confusion and henon map diffusion. *Adv. Sci. Technol. Eng. Syst.*, 2021.

[99] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, and Elaine Barker. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, Booz-allen and hamilton inc mclean va, 2001.

[100] Ahmed A Abd El-Latif, Li Li, and Xiamu Niu. A new image encryption scheme based on cyclic elliptic curve and chaotic system. *Multimedia tools and applications*, 2014.

[101] Zhi-liang Zhu, Wei Zhang, Kwok-wo Wong, and Hai Yu. A chaos-based symmetric image encryption scheme using a bit-level permutation. *Information Sciences*, 2011.

PUBLICATIONS

Gaetan Gildas Gnyamsi Nkuigwa,Hermann Djeugoue Nzeuga, J.S.Armand Eyebe Fouda, Samrat L.Sabat, Wolfram Koepf. An extendable key space integer image-cipher using 4-bit piece-wise linear cat map. *Multimedia Tools and Applications*, 1-23, 2022

Multimedia Tools and Applications
https://doi.org/10.1007/s11042-022-13779-y

An extendable key space integer image-cipher using 4-bit piece-wise linear cat map

Gaetan Gildas Gnyamsi Nkuigwa[1] · Hermann Djeugoue Nzeuga[1] · J. S. Armand Eyebe Fouda[1,2] · Samrat L. Sabat[3] · Wolfram Koepf[2]

Received: 10 September 2021 / Revised: 30 June 2022 / Accepted: 5 September 2022

Abstract
This paper presents a multiplierless image-cipher, with extendable 2048-bit key-space, based on a 4-dimensional (4D) quantized piece-wise linear cat map (PWLCM). The quantized PWLCM exhibits limit-cycles of 4-bit encoded integers with periods greater than 10^7. The synthesis of the PWLCM in a finite state space allows to eliminate the undesirable finite precision effect due to the hardware realization. The proposed image-cipher combines chaos, modular arithmetic, and lattice-based cryptography to encrypt a color image by performing pixel permutation and diffusion in a single operation. Further, an image-dependent confusion operation based on an 8-bit 2D-PWLCM is performed on the whole image to enhance security. In order to increase the key-space without key duplication, 16×16 sub-images are modified using sub-keys of different lattice length vectors generated from the external key. Both simulations and security analyses confirm that the proposed algorithm can resist common cipher attacks, in addition to its advantages such as simplicity, ease of implementation on low-end processors and extensibility of key-space that allows it to easily adapt even for future post-quantum computing attacks.

Keywords Cryptography · Modular arithmetic · Random integers · Security analysis

1 Introduction

The rapid development of digital image transmission over wireless communication media has increased the concern of data security, leading to the demand for image cipher. There exist several techniques for securing data, including steganography, watermarking, and data encryption [2, 25]. The steganography technique conceals the message data; hence it requires more communication bandwidth over a computer network, whereas the cryptography technique transforms the message data, needing approximately the same bandwidth as message data. Thus, encryption is the preferable and mature technology used in applications

✉ J. S. Armand Eyebe Fouda
fouda@mathematik.uni-kassel.de

Extended author information available on the last page of the article.

involving message transmission over a network [19]. In encryption, the message is scrambled using a secret key. The encryption's strength depends on the secret key's randomness strength. Different techniques such as linear congruential, additive congruentional, linear feedback shift register, multiple recursive generators and chaos based generators are used to generate the random sequences using the seed as the initial condition of the dynamical system that constitutes the secret key [18, 24, 37].

There exist different methods such as linear feedback shift register (LFSR), linear congruential generator (LCG), Multiple Recursive Generators (MRGs) for generating Pseudo-random Numbers (PRN). The disadvantages of these methods are the limited periodicity of the random number sequence, which can be mitigated using a suitable chaotic system.

Chaotic systems have intrinsic properties such as sensitivity to initial conditions, ergodicity, random-like dynamics behaviors, and unpredictability that are desired characteristics for designing secure ciphers. Hence, chaos-based encryption methods have emerged besides traditional ciphers such as Advanced Encryption Standard (AES), International Data Encryption Algorithm (IDEA), Data Encryption Standard (DES), and RSA to enhance data security. However, complex chaotic systems are required to make ciphers more secure. Broadly, there are two types of dynamical chaotic systems: continuous-time and discrete-time. The latter is suitable for data encryption as it is feasible to implement on digital hardware. Among the basic discrete-time systems, 2D logistic map, 2D standard map, 2D Henon map, and the 3D Baker map are used in cryptography [26, 34, 38, 39]. During the last decade, the performance of such systems has been improved to increase their complexity, leading to more randomness for secure data encryption. A combination of the piece-wise-linear chaotic map and linear Diophantine Equation (LDE) enhances the cipher's security and was used for image encryption [14]. Although the ciphered image was obtained after only one round, it has the drawback that the encryption process is independent of the plain-image characteristics. The authors mitigated the drawbacks by proposing another one-round encryption scheme in which large permutation and diffusion keys were generated by sorting the solutions of the LDE [13]. Multiple chaotic maps were used to derive the control parameters and initial values to increase the security level of the cipher [31]. It enhances the key-space; however, the short-length encryption key makes the cipher vulnerable against brute force attacks. Other ciphers based on the combination of chaotic systems were also proposed to increase the key-space and are still under investigation [22, 29, 30, 33, 36].

In all the above chaos-based ciphers, chaotic maps need to be quantized during the hardware implementation of the algorithm. Such a quantization reduces the randomness of the chaotic orbits; hence the security level of the cipher [41]. In order to overcome such a drawback, it is necessary to either evaluate the complexity of the chaotic system under limited precision conditions or to increase the computational precision. Most of the work reported in the literature implements chaotic systems with 32-bit floating-point arithmetic, which is hardware costly and requires high-end processors for execution [5, 15]. Hence, it is necessary to design quantized chaotic systems with a large period of limit-cycles to implement chaos-based ciphers on a low-end processor [12].

Arnold's cat map (ACM) preserves the mixing property even after quantization among the different chaotic maps. It has the advantage of (i) being easily defined both in the continuous phase space and the discrete phase space (quantized version) and (ii) a computationally simple 2D system that can be easily extended into a multi-dimensional system. It has been

used in data encryption in the confusion step. Apart from the ACM, the combination of the Henon and Arnold cat maps was used in designing an image cipher [11]. Although the algorithm performed well, it has nevertheless been observed that the periodicity of restoring the original image is too short, leading to security issues. More recently, an encryption scheme based on continuous phase space of a generalized Arnold cat map was reported [20]. In [43], the 2D ACM was combined with an affine cipher to enhance the security level of the cipher. In all of these algorithms, the ACM is used in its continuous phase space version, and the precision of the generated obits is chosen as large as possible (32-bit, for example) to avoid short limit-cycles due to the quantization process. The quantized ACM (QACM) has been widely studied in the literature, and the relationship between its period and the number of encoding bits has been determined [6, 7, 27].

The period of the QACM is an important parameter that can induce security issues when using it in cryptography. It is known that the period of the QACM does not exceed $3m$, $m \in \mathbb{N}_{>1}$ being the modulo value. This limitation justifies the use of continuous phase space ACM ($m = 1$) compared to QACM in many cryptographic applications. As a solution, some researchers investigated the impact of the dimensionality and the control parameters on the period of the system. The investigation showed that the period does not significantly increase with the increase in dimension [16], where the conventional 2D Arnold's cat map was altered to a 3D map by introducing six control parameters. The obtained map allows for improvement, but the period distribution and its impact on the system dynamics are not evaluated. Due to the finite computer precision, chaotic sequences are transformed into periodic ones. Further, the output of the 3D map was perturbed to mitigate such degradation of chaotic sequences without investigating the impact of the perturbation on randomness.

The present paper proposes a multiplierless 2048-bit key secure cipher based on the quantized piece-wise linear cat map (PWLCM) obtained by perturbing the conventional QACM [11]. We aim to directly generate randomly distributed integers with the desired precision using the PWLCM. The proposed algorithm combines chaos, modular arithmetic, and lattice-based cryptography [3, 8, 35]. The latter allows to easily extend the external key length without duplication. Such a property is required as lattice-based ciphers are assumed to resist future attacks in the era of post-quantum computing [3, 17, 28, 35]. For the algorithm to be implemented even with low-end processors, we consider only 4-bit precision random numbers generated from a 4D PWLCM for performing the confusion and diffusion operations. We investigate the period of the generated random integers and their randomness to prove the high-security level of our cipher. In the proposed scheme, pixel positions and values are modified in a single operation within blocks of size 16×16 pixels with a 2D PWLCM before confusing the whole image. It helps to enhance the speed performance. During the confusion-diffusion operation, each 16×16 sub-image is modified using a different subkey corresponding to a combination of κ-length vectors (κ-dimensional lattice), $\kappa \in \mathbb{N}_{\geq 1}$ [28].

The key contributions of the current work are as follows:

1. A novel integer arithmetic multiplierless 2048-bit key image cipher combining chaos, modular arithmetic and lattice-based cryptography is proposed that uses a combination of 4-bit and 8-bit modular addition and subtraction operations only;
2. An extensible key management technique combining modular arithmetic and lattices is presented;

3. A 4-bit 2D and 4D PWLCM is proposed to generate large period pseudorandom sequences;
4. A performance analysis for different attacks is presented.

The rest of the paper is organized as follows: Section 2 presents a brief recalling of ACM and the definition of the PWLCM; Section 3 presents the proposed cipher; Section 4 presents the security analysis of the proposed encryption scheme, and conclusions are given in Section 5.

2 The 4D piece-wise linear cat map (PWLCM)

2.1 Brief recall on 2D PWLCM

Arnold's cat map is basically a 2D chaotic map of repeated folding and stretching in a limited area. It has been popularly used in multimedia chaotic encryption [5]. The 2D ACM is modeled as [23]:

$$\begin{cases} x(t+1) = x(t) + \alpha y(t) \\ y(t+1) = \beta x(t+1) + y(t) \end{cases} \bmod m, \tag{1}$$

which can be rewritten using matrix representation as

$$\mathbf{x}(t+1) = A\mathbf{x}(t) \bmod m \tag{2}$$

where

$$A = \begin{pmatrix} 1 & \alpha \\ \beta & \alpha \cdot \beta + 1 \end{pmatrix},$$

$(\alpha, \beta) \in \mathbb{N}_{\geq 1}^2$, and $\mathbf{x} = (x, y)^T$; $(\cdot)^T$ is the transpose of $(\cdot)$. The above map is a discrete time system and is continuous in the phase space for $(x, y) \in [0, 1)^2$ and $m = 1$. The QACM is obtained for $(x, y) \in [0, m)^2$ with $m \in \mathbb{N}_{>1}$. The QACM is periodic and its period depends on m and the parity of both α and β. It is shown that for $\alpha = \beta = 1$ and $m = 2^n$, the period Π_n behaves like [4, 10]

$$\Pi_n = 2 \cdot \Pi_{n-1},\ n > 2 \tag{3}$$

with $\Pi_1 = \Pi_2 = 3$ for the minimal period.

The period of the QACM can be computed using (3) and it is too short. As an example, for an 8-bit encoded phase space values, the period is $\Pi_8 = 192$. In order to increase the period of QACM, we proposed the PWLCM by introducing a nonlinear perturbation term to the conventional QACM, as shown below [9]:

$$\mathbf{x}(t+1) = A\mathbf{x}(t) + \mathbf{x}_c(t) \bmod m, \tag{4}$$

where the perturbation $\mathbf{x}_c(t)$ is defined as

$$\mathbf{x}_c(t) = \begin{pmatrix} \sum_{i=1}^{M} \left(a_i + y(t)\right) \bmod c_i \\ \sum_{j=1}^{N} \left(b_j + x(t+1)\right) \bmod d_j \end{pmatrix} \tag{5}$$

with $(i, j) \in \mathbb{N}$. In (4), c_i and d_j are two natural numbers such that $0 \leq c_i < m + a_i$ and $0 \leq d_j < m + b_j$, $0 \leq a_i, b_j < m$ if $(c_i, d_j) = (0, 0)$; $0 \leq a_i < c_i$ if $c_i \neq 0$ and, $0 \leq b_j < d_j$ if $d_j \neq 0$. The parameters a_i, b_j, c_i and d_j are defined as perturbation parameters that can also be used as control parameters, while $\mathbf{a} = (a_i)$, $\mathbf{b} = (b_j)$, $\mathbf{c} = (c_i)$

and $\mathbf{d} = (d_j)$ are control vectors. We showed in [9] that the system in (5) can be put in the form

$$\mathbf{x}(t+1) = B\mathbf{x}(t) + C(t) \mod m \tag{6}$$

where

$$B = \begin{pmatrix} 1 & \alpha' \\ \beta' & \alpha'\beta' + 1 \end{pmatrix}, \tag{7}$$

with $\alpha' = M + \alpha$, $\beta' = N + \beta$ and $C(t) = \Big(C_1(t), C_2(t)\Big)^T$ such that

$$C(t) = \begin{pmatrix} \sum_{i=1}^{M} \Big(a_i - \varepsilon_i c_i \cdot u\Big(a_i + y(t) - c_i\Big)\Big) \\ \beta' C_1(t) + \sum_{j=1}^{N} \Big(b_j - \varepsilon_j d_j \cdot u\Big(b_j + x(t+1) - d_j\Big)\Big) \end{pmatrix}, \tag{8}$$

where $\varepsilon_i = \left\lfloor \frac{a_i + y(t)}{c_i} \right\rfloor$ and $\varepsilon_j = \left\lfloor \frac{b_j + x(t+1)}{d_j} \right\rfloor$. $u(t)$ is the Heaviside function defined as

$$u(t) = \begin{cases} 0, & \text{if } t < 0; \\ 1, & \text{otherwise.} \end{cases} \tag{9}$$

We verified that the PWLCM is a conservative system that exhibit large periods [9]. In order to use it both for image scrambling and diffusion, we suggest its 4D modelling.

2.2 The proposed 4D PWLCM

The above 2D PWLCM can be easily extended to a 4D PWLCM by coupling two 2D PWLCM $\mathbf{x} = (x, y)^T$ and $\mathbf{z} = (q, r)^T$ such that:

$$\begin{cases} x(t+1) = x(t) + \alpha y(t) + F_1(y, t) \\ y(t+1) = y(t) + \beta x(t+1) + F_2(x, t) \\ q(t+1) = q(t) + \gamma y(t+1) + F_3(y, t) \\ r(t+1) = r(t) + \zeta q(t+1) + F_4(q, t) \end{cases} \mod m, \tag{10}$$

where $(\alpha, \beta, \gamma, \zeta) \in \mathbb{N}^4$. $F_1(y, t)$, $F_2(y, t)$, $F_3(y, t)$ and $F_4(y, t)$ are the coupling nonlinear terms, defined as

$$\begin{cases} F_1(y, t) = \sum_{i=1}^{M} \Big(a_i + y(t)\Big) \mod c_i \\ F_2(x, t) = \sum_{j=1}^{N} \Big(b_j + x(t+1)\Big) \mod d_j \\ F_3(y, t) = \sum_{k=1}^{P} \Big(e_k + y(t+1)\Big) \mod g_k \\ F_4(q, t) = \sum_{l=1}^{W} \Big(f_l + q(t+1)\Big) \mod h_l \end{cases} \tag{11}$$

The 4D PWLCM defined in (10) is invertible and the corresponding inverse system is

$$\begin{cases} r(t) = z(t+1) - \zeta q(t+1) - F_4(q, t) \\ q(t) = q(t+1) - \gamma y(t+1) - F_3(y, t) \\ y(t) = y(t+1) - \beta x(t+1) - F_2(x, t) \\ x(t) = x(t+1) - \alpha y(t) - F_1(y, t) \end{cases} \mod m. \tag{12}$$

2.3 Stability analysis

In order to investigate the stability of the 4D PWLCM, we evaluated its Jacobian matrix. By using the same expansion as in (5)–(9), the system can be rewritten as in (6) with

$$B = \begin{pmatrix} 1 & \alpha' & 0 & 0 \\ \beta' & 1+\alpha'\beta' & 0 & 0 \\ \beta'\gamma' & \gamma'(1+\alpha'\beta') & 1 & 0 \\ \beta'\gamma'\zeta' & \gamma'\zeta'(1+\alpha'\beta') & \zeta' & 1 \end{pmatrix}, \tag{13}$$

and $C(t) = \Big(C_1(t), C_2(t), C_3(t), C_4(t)\Big)^T$ such that

$$C(t) = \begin{pmatrix} \sum_{i=1}^{M}\Big(a_i - \varepsilon_i c_i \cdot u\Big(a_i + y(t) - c_i\Big)\Big) \\ \beta' C_1(t) + \sum_{j=1}^{N}\Big(b_j - \varepsilon_j d_j \cdot u\Big(b_j + x(t+1) - d_j\Big)\Big) \\ \gamma' C_2(t) + \sum_{k=1}^{P}\Big(e_k - \varepsilon_k g_k \cdot u\Big(e_k + y(t+1) - g_k\Big)\Big) \\ \zeta' C_3(t) + \sum_{l=1}^{W}\Big(f_l - \varepsilon_l h_l \cdot u\Big(f_l + q(t+1) - h_l\Big)\Big) \end{pmatrix}, \tag{14}$$

From the above equations, we deduce the Jacobian matrix as

$$J = \begin{pmatrix} 1 & J_{1,2}(t) & 0 & 0 \\ J_{2,1}(t) & J_{2,2}(t) & 0 & 0 \\ J_{3,1}(t) & J_{3,2}(t) & 1 & 0 \\ J_{4,1}(t) & J_{4,2}(t) & J_{4,3}(t) & 1 \end{pmatrix}, \tag{15}$$

where

$$\begin{cases} J_{1,1}(t) = \alpha' - \sum_{i=1}^{M}(c_i\delta(y(t) + a_i - c_i)) \\ J_{2,1}(t) = \beta' - \sum_{j=1}^{N}(d_j\delta(x(t+1) + b_j - d_j)) \\ J_{2,2}(t) = 1 + J_{2,1}(t)J_{1,2}(t) \\ J_{3,1}(t) = J_{2,1}(t)(\gamma' - \sum_{k=1}^{P}(g_k\delta(y(t+1) + e_k - g_k))) \\ J_{3,2}(t) = J_{2,2}(t)(\gamma' - \sum_{k=1}^{P}(g_k\delta(y(t+1) + e_k - g_k))) \\ J_{4,1}(t) = J_{3,1}(t)J_{4,3}(t) \\ J_{4,2}(t) = J_{3,2}(t)J_{4,3}(t) \\ J_{4,3}(t) = \zeta' - \sum_{l=1}^{W}(h_l\delta(q(t+1) + f_l - h_l)) \end{cases},$$

and $\alpha' = \alpha + M$, $\beta' = \beta + N$, $\gamma' = \gamma + P$, $\zeta' = \zeta + W$. The 4D PWLCM thus defined is conservative as $\det(J) = 1$, which implies that the sum of the four corresponding Lyapunov exponents is equal to 0. While computing the eigenvalues of J, we found $\Lambda_1 = \Lambda_2 = 1$, the other eigenvalues are

$$\Lambda_3(t) = 1 + \frac{1}{2}\Big(\alpha' - \sum_{i=1}^{M} c_i\delta\Big(y(t) - \tau_y^i\Big)\Big)\Big(\beta' - \sum_{j=1}^{N} d_j\delta\Big(x(t+1) - \tau_x^j\Big)\Big)$$
$$\left(1 + \sqrt{1 + \frac{4}{\Big(\alpha' - \sum_{i=1}^{M} c_i\delta\Big(y(t) - \tau_y^i\Big)\Big)\Big(\beta' - \sum_{j=1}^{N} d_j\delta\Big(x(t+1) - \tau_x^j\Big)\Big)}}\right) \tag{16}$$

and

$$\Lambda_4(t) = 1 + \frac{1}{2}\left(\alpha' - \sum_{i=1}^{M} c_i \delta\left(y(t) - \tau_y^i\right)\right)\left(\beta' - \sum_{j=1}^{N} d_j \delta\left(x(t+1) - \tau_x^j\right)\right) \left(1 - \sqrt{1 + \frac{4}{\left(\alpha' - \sum_{i=1}^{M} c_i \delta\left(y(t) - \tau_y^i\right)\right)\left(\beta' - \sum_{j=1}^{N} d_j \delta\left(x(t+1) - \tau_x^j\right)\right)}}\right) \tag{17}$$

The Lyapunov exponents corresponding to $\Lambda_{1,2}$ are equal to zero. The sum of the Lyapunov exponents being zero implies that $\Lambda_3 > 1$ (corresponding to a positive Lyapunov exponent) and $0 < \Lambda_4 < 1$ (corresponding to a positive Lyapunov exponent). The steady state of the system depends on the perturbing parameter and remains difficult to formally determine. Figure 1 presents the behavior of the largest Lyapunov exponent for arbitrary parameter setting and various initial conditions (x_0, y_0, q_0, r_0). We fixed $q_0 = r_0 = 1$ and set $z_0 = 2^n x_0 + y_0$, where $0 \leq x, y \leq 2^n - 1$, $a_r = \sum_{i=1}^{N} 2^{n(i-1)} a_N$, and $n = 4$ is the number of bits or precision. Figure 1(a) shows the Lyapunov exponent as a function of initial conditions z_0, where control vectors are set as $\mathbf{a} = \mathbf{e} = (1, 1, 0, 0)$, $\mathbf{b} = \mathbf{f} = (0, 2, 0, 0)$, $\mathbf{c} = \mathbf{g} = (0, 3, 1, 1)$, $\mathbf{d} = \mathbf{h} = (3, 5, 1, 1)$. Figure 1(b) depicts the Lyapunov exponent in terms of the control vector $\mathbf{a}$ represented as parameter a_r, with $N = 3$ where $x_0 = y_0 = 2$, $\mathbf{b} = (0, 2, 1)$, $\mathbf{c} = (15, 15, 15)$, $\mathbf{d} = (3, 5, 1)$, $\mathbf{e} = (1, 1, 0)$, $\mathbf{f} = (0, 2, 0)$, $\mathbf{g} = (0, 3, 1)$, and $\mathbf{h} = (3, 5, 1)$. These plots show that the largest Lyapunov exponent remains positive for the chosen initial conditions and control parameters.

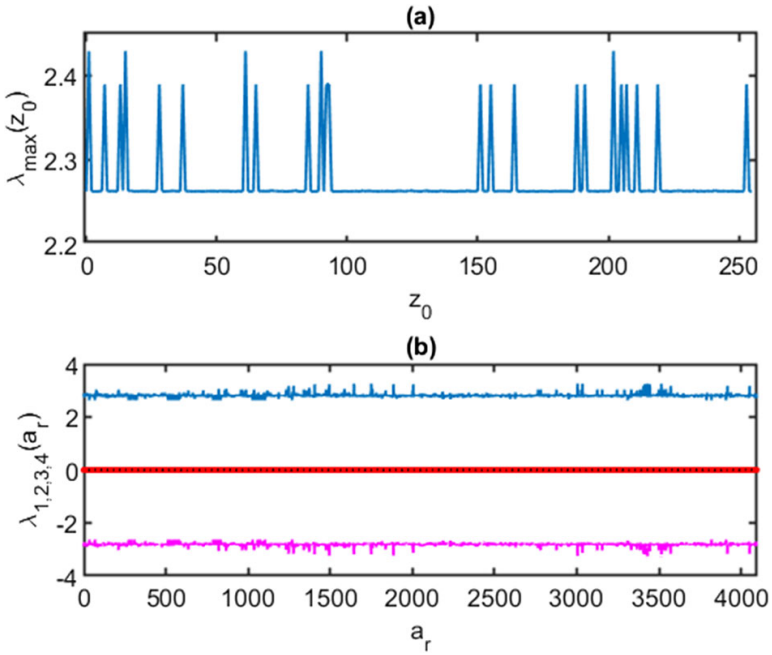


Fig. 1 Lyapunov exponent of the 4D PWLCM: (a) behavior of the Lyapunov exponent $\lambda(z_0)$ with respect to the initial condition $z_0 = 2^n x_0 + y_0$, for $n = 4$ and given control vectors; (b) behavior of the four Lyapunov exponents $\lambda_{1,2,3,4}(a_r)$ with respect to the control vector $\mathbf{a}$, for $n = 4$ and given initial condition and control vectors $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d}$, $\mathbf{e}$, $\mathbf{f}$, $\mathbf{g}$, and $\mathbf{h}$

Table 1 Comparison of the 2D QACM and 2D PWLCM periods

n	(a_1, b_1)	(c_1, d_1)	Π_{QACM}	Π_{PWLCM}
2	(1,2)	(3,3)	3	90
3	(3,3)	(2,3)	6	780
4	(5,1)	(11,3)	12	12759390
5	(2,2)	(7,5)	24	56934108
6	(6,2)	(5,13)	48	1.7870e+12
7	(2,2)	(7,5)	96	1.2041e+16
8	(6,2)	(5,13)	192	6.918e+56

2.4 Period and randomness evaluation of the PWLCM

Similar to the QACM, the proposed 4D PWLCM is chaotic while used in a continuous phase space ($m = 1$). As we are interested in using it in a finite state phase space ($m = 4$ and $m = 8$), it is no longer chaotic but preserves the mixing properties of the corresponding chaotic systems. For it to be efficient for security applications, its period should be very large. In this subsection, we estimate the period Π_n of the proposed 4D PWLCM for different values of the precision n and some arbitrary values of the perturbation parameters. The periods of PWLCM are compared with the QACM and tabulated in Tables 1 and 2. From the Table 1, we can observe that for any value of n, the period of the proposed PWLCM is significantly higher than that of the corresponding QACM.

Table 2 shows the comparison of the 4D PWLCM and 4D QACM. It confirms that the periods of the proposed map are significantly higher than those of the QACM for any arbitrary parameter values, $\alpha = \beta = \gamma = \zeta = 1$, $M = N = 1$:

In order to testify the mixing property of the 4D PWLCM, we propose to shuffle a $2^{13} \times 2^{13}$ periodic image obtained by repeating sequences of 8-bit encoded unsigned integers. The image was shuffled with x and y coordinates of the 4D PWLCM. We set as initial conditions, $q_0 = 1$, $r_0 = 2$; $x_0, y_0 \in \left[0, 2^{13} - 1\right]^2$ and controls parameters $a_1 = 1, b_1 = 2, e_1 = 0, f_1 = 1, c_1 = 3, d_1 = 3, g_1 = 3$ and $h_1 = 3$; the corresponding period is 148740480. The NIST800-22 statistical test was performed after 30 iterations of image scrambling with the PWLCM and the QACM. The data set was divided into 100 sets of 1000000 bits and the results obtained are summarized in Table 3. The comparison of the two results confirms that the 4D PWLCM is suitable for the image scrambling as it passes all the tests.

Table 2 Comparison of the 4D QACM and 4D PWLCM periods

n	(a_1, b_1, e_1, f_1)	(c_1, d_1, g_1, h_1)	Π_{QACM}	Π_{PWLCM}
2	(1,2,1,0)	(3,3,3,3)	12	360
3	(6,3,1,1)	(11,1,5,11)	24	240240
4	(6,3,1,1)	(7,1,1,1)	48	340728960

Table 3 NIST 800-22 test results

Sub-Tests	4D QACM		4D PWLCM	
	P-value	Proportion	P-value	Proportion
Frequency	0.0	22/100	0.319084	99/100
Block frequency	0.0	0/100	0.574903	98/100
Cumulative sums (forward)	0.0	0/100	0.289667	100/100
Runs	0.0	0/100	0.657933	99/100
Longest run	0.0	0/100	0.017912	100/100
Rank	0.0	0/100	0.657933	100/100
FFT	0.0	0/100	0.319084	98/100
Non overlapping	0.0	0/100	0.971699	100/100
Overlapping	0.0	0/100	0.574903	99/100
Universal	0.0	0/100	0.066882	100/100
Approximate entropy	0.0	0/100	0.304126	100/100
Random excursions	0.0	0/61	0.957319	61/61
Random excursions variant	0.0	0/61	0.957319	61/61
Serial	0.0	0/100	0.955835	98/100
Linear complexity	0.0	0/100	0.955835	100/100

3 Proposed encryption algorithm

The proposed encryption algorithm has two stages, namely diffusion-confusion and confusion only. In the pixel diffusion-confusion stage, the pixel value of each sub-block is diffused and confused using the 4D PWLCM, whereas in the subsequent block confusion stage, each diffusion-confusion sub-block is split into sub-images that are confused using the 2D PWLCM.

The proposed scheme generates the initial values and the control parameter values of 4D PWLCM from the external key, whereas the control parameters of the 2D PWLCM is derived from the external key along with the diffusion-confusion image. Thus, it is image-dependent. The detailed procedure for generating these parameters is described in Section 3.1.

The algorithmic steps of the proposed cipher are mentioned below:

The block diagram of the proposed image cipher is shown in Fig. 2. The minimum number of rounds for the algorithm to be secure is $R = 2$. Indeed, once the image-dependent step is applied in the first round, we need to go for a second round for the image-dependent shuffling to take effect in the diffusion process, thus increasing the algorithm's sensitivity to the plain-image.

3.1 External key management

The external key is defined by using N_K ASCII characters, $\{C_k\}$, $0 \leq k \leq N_K - 1$, to derive κ-length ($\kappa = 2\lfloor \frac{N_K}{8} \rfloor$) control vectors **a**, **b**, **c**, **d**, **e**, **f**, **g**, **h** whose coordinates are 4-bit encoded unsigned integers. As ASCII characters are 8-bit encoded, each character is divided into two blocks of 4 bits that are used as coordinates of each control vector. Therefore, $\theta = \frac{\kappa}{2}$ ASCII characters are required to determine the coordinates of each control vector. In

1. Consider an $N_L \times N_C$ image and reshape it into a 1D image.
2. Split the 1D image into blocks of length 1×256; for 16×16 sub-block.
3. Set the external key and deduce control vectors **a**, **b**, **c**, **d**, **e**, **f**, **g**, **h** as κ-length vectors of 4-bit encoded integers as shown in Section 3.1.
4. Set the control vector coordinates corresponding to the sub-image rank as detailed in Section 3.3. Define the corresponding 4D PWLCM as in (22).
5. Decompose the actual pixel into two 4-bit encoded integers $0 \leq q, r \leq 15$ as detailed in Section 3.2.
6. Apply the 4D PWLCM to each pixel by using as input its position referenced as $0 \leq x, y \leq 15$ and its gray-level value represented by $0 \leq q, r \leq 15$.
7. Repeat steps 5-6 until the whole sub-image is modified and consider the next sub-image.
8. Repeat steps 5-7 until the whole image is modified.
9. Use the image obtained from step 8 to define an image-dependent key by defining the 2D PWLCM control parameters as detailed in Section 3.4.
10. Confuse the block positions by using the image dependent 2D PWLCM to obtain a one round modified image.
11. Consider the previous image and go to step 4 for another round.

Algorithm 1

the case of 256-bit key for example, $\theta = 4$ and characters C_0 to C_3 are used to determine coordinates of the control vector **a**, C_4 to C_7 are used for **b**, C_8 to C_{11} for **c**, C_{12} to C_{15} for **d**, C_{16} to C_{19} for **e**, C_{20} to C_{23} for **f**, C_{24} to C_{27} for **g** and C_{28} to C_{31} are used for **h**. In the case of a 2048-bit key, $\theta = 64$ ASCII characters are required to determine each control vector. Therefore, **a** for example is defined as:

$$\begin{cases} \mathbf{a}(2\xi - 1) = \left\lfloor \frac{C_{\xi-1}}{16} \right\rfloor \\ \mathbf{a}(2\xi) = C_{\xi-1} \mod 16 \end{cases} \tag{18}$$

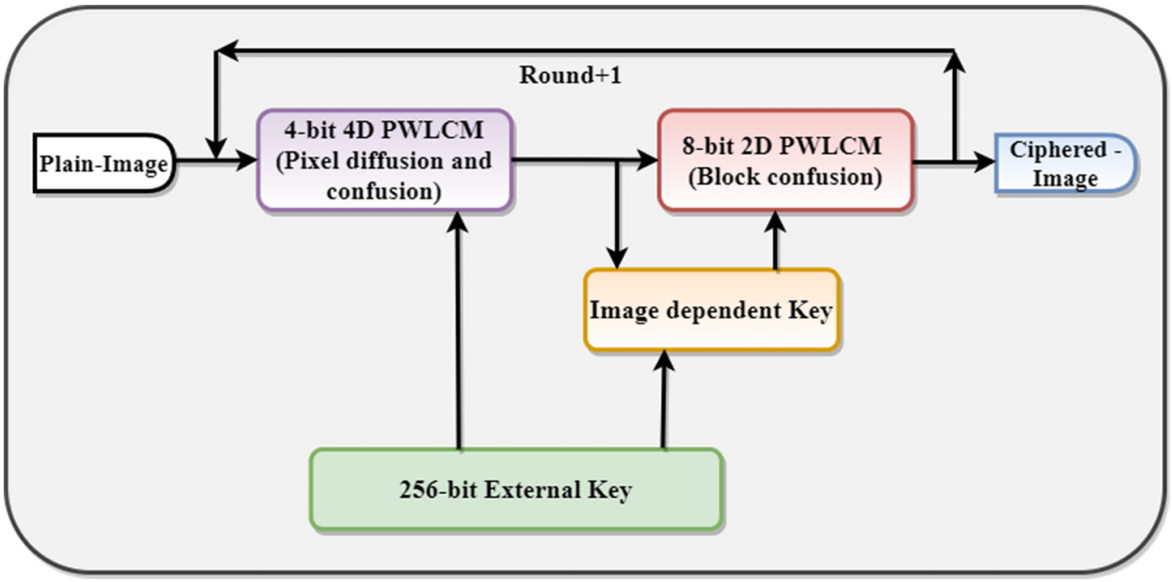


Fig. 2 Synoptic of the proposed cipher

where $1 \leq \xi \leq \theta$. The other vectors **b**, **e** and **f** are defined using the same principle, from the corresponding ASCII symbols. Similarly, **c** is defined as:

$$\begin{cases} \mathbf{c}(2\xi - 1) = \mathbf{a}(2\xi - 1) + \left\lfloor \frac{C_{\xi+\kappa-1}}{16} \right\rfloor \\ \mathbf{c}(2\xi) = \mathbf{a}(2\xi) + \left(C_{\xi+\kappa-1} \mod 16 \right) \end{cases} \tag{19}$$

where $1 \leq \xi \leq \theta$. The coordinates of the other control vectors **d**, **g** and **h** can be defined using the same approach. Thus, control vectors can be considered as belonging to an κ-D lattice.

3.2 Pixel decomposition

In step 5, individual pixel, i of sub-image S_j, $j \in \mathbb{N}$ are decomposed into two 4-bit encoded integers q_i and r_i. For an 8-bit encoded pixel P_i, q_i and r_i are obtained as

$$q_i = \left\lfloor \frac{P_i}{16} \right\rfloor.$$

and

$$r_i = P_i \mod 16$$

3.3 Pixel confusion-diffusion process

In step 6, the confusion and diffusion operations are combined in a single operation (confusion-diffusion). Indeed, the coordinates x_i and y_i, as well as the intensity coordinates q_i and r_i of the pixel P_i are used as initial conditions of the 4D PWLCM to output new coordinates $x_{i'}$, $y_{i'}$, $q_{i'}$ and $r_{i'}$ using (20). As S_j is a vector, there is a relationship between x_i, y_i and i such that

$$x_i = i \mod 16,$$

and

$$y_i = \left\lfloor \frac{i}{16} \right\rfloor.$$

For each sub-image S_j, only a single coordinate $\mathbf{a}(k)$, $\mathbf{b}(k)$, $\mathbf{c}(k)$, $\mathbf{d}(k)$, $\mathbf{e}(k)$, $\mathbf{f}(k)$, $\mathbf{g}(k)$, and $\mathbf{h}(k)$, $k > 0$, is used as control parameter to 4D PWLCM as given below:

$$\begin{cases} x(t+1) = x(t) + \alpha y(t) + (\mathbf{a}(k) + y(t)) \mod \mathbf{c}(k) \\ y(t+1) = y(t) + \beta x(t+1) + (\mathbf{b}(k) + x(t+1)) \mod \mathbf{d}(k) \\ q(t+1) = q(t) + \gamma y(t+1) + (\mathbf{e}(k) + y(t+1)) \mod \mathbf{g}(k) \\ r(t+1) = r(t) + \zeta q(t+1) + (\mathbf{f}(k) + q(t+1)) \mod \mathbf{h}(k) \end{cases} \mod 16, \tag{20}$$

where $k = 1 + j \bmod \kappa$. The new position of the diffused pixel i' is obtained after three iterations of the PWLCM as

$$i' = 16 \cdot y_{i'} + x_{i'} \tag{21}$$

The corresponding intensity value is obtained as

$$P_{i'} = 16 \cdot q_{i'} + r_{i'}. \tag{22}$$

3.4 Image-dependent confusion

In order to enhance the security level of the cipher and prevent chosen-plaintext attacks, an additional image-dependent confusion step is performed using the 2D PWLCM, described as

$$\begin{cases} x(t+1) = \left(x(t) + \alpha y(t) + \sum_{k=1}^{16} \left(\mathbf{a_1}(k) + y(t) \right) \bmod \mathbf{c_1}(k) \right) \bmod m_1 \\ y(t+1) = \left(y(t) + \beta x(t+1) + \sum_{k=1}^{16} \left(\mathbf{b_1}(k) + x(t+1) \right) \bmod \mathbf{d_1}(k) \right) \bmod m_2 \end{cases} \tag{23}$$

where $\mathbf{a_1}$, $\mathbf{b_1}$, $\mathbf{c_1}$ and $\mathbf{d_1}$ are 16-length control vectors whose coordinates are 4-bit encoded values derived from the image of step 8 and the external key as:

$$\begin{cases} \mathbf{a_1}(1:\kappa) = \Gamma \mod \mathbf{c} \\ \mathbf{a_1}(\kappa+1:2\kappa) = \Gamma \mod \mathbf{d} \\ \mathbf{b_1}(1:\kappa) = \Gamma \mod \mathbf{g} \\ \mathbf{b_1}(\kappa+1:2\kappa) = \Gamma \mod \mathbf{h} \\ \mathbf{c_1}(1:\kappa) = \mathbf{a_1}(1:\kappa) + \Gamma \mod \mathbf{a} \\ \mathbf{c_1}(\kappa+1:2\kappa) = \mathbf{a_1}(\kappa+1:2\kappa) + \Gamma \mod \mathbf{b} \\ \mathbf{d_1}(1:\kappa) = \mathbf{b_1}(1:\kappa) + \Gamma \mod \mathbf{e} \\ \mathbf{d_1}(\kappa+1:2\kappa) = \mathbf{b_1}(\kappa+1:2\kappa) + \Gamma \mod \mathbf{f} \end{cases} \tag{24}$$

where

$$\Gamma = \sum_{i=1}^{N_L} \sum_{j=1}^{N_C} I_c(i, j). \tag{25}$$

I_c is the intermediate ciphered image obtained in step 8. m_1 and m_2 are defined such that

$$\begin{cases} m_1 = \frac{N_L}{T_1} \\ m_2 = \frac{N_C}{T_2} \end{cases} \tag{26}$$

with $(T_1, T_2) \in \mathbb{N}^2_{\geq 1}$. $T_1 \times T_2$ is the size of sub-images to be shuffled, $m_1 \times m_2$ is the number of sub-images and $N_L \times N_C$ is the size of the image.

4 Results and security analysis

The performance of the proposed encryption algorithm is analyzed by encrypting standard images like "Lena", "Baboon", "Airplane", "Peppers", of size 512 × 512 and 256 gray levels. Figure 3 respectively represents the plain-text "Lena" image, its encrypted image, and the decrypted image with the same key. All simulations are performed using MATLAB 2018b on a CPU with an Intel(R) Core (TM) i5-8250u CPU @ 1.60 GHz and 8 GB RAM with the Windows 10 operating system. In the current simulation, a 256-bit external encryption key is set as $K1 = azertyuiopqsdfgjazertyuiopqsdfg0$. We also set $\alpha = \beta = \gamma = \zeta = 1$ to make the algorithm multiplierless.

4.1 Evaluation metrics

This subsection presents the definition of evaluation metrics used to measure the proposed cipher's strength.

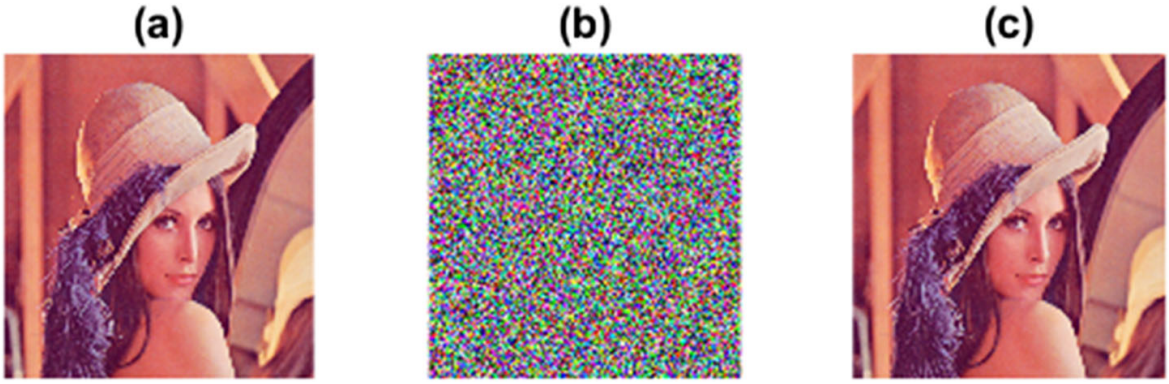


Fig. 3 Example of ciphered image of Lena using the proposed encryption algorithm: (a) plain-text image; (b) ciphered image and (c) decrypted image

4.1.1 Entropy measure

The Shannon entropy of the ciphered image is the primary indicator to confirm that the cipher is secure against permutation of pixels. It measures the disorder or randomness of pixels. For an 8-bit encoded image, it is determined as

$$H = -\sum_{i=0}^{255} p(v_i) \log_2(p(v_i)), \tag{27}$$

where $0 \leq v_i \leq 255$ are pixel values and $p(v_i)$ the probability of v_i. It is to be noted that for an 8-bit encoded image, the maximum value of the entropy is $H = 8$.

4.1.2 Correlation coefficient

The correlation coefficient is used to measure the similarity between two images A and B, and is defined as

$$\rho_{A,B} = \frac{E\Big((A - \overline{A})(B - \overline{B})\Big)}{\sigma_A \cdot \sigma_B}, \tag{28}$$

where $E(\cdot)$ is the expectation value; A and B are images between which the correlation coefficient needs to be evaluated. $\overline{A}$ and $\overline{B}$ represent the mean value of images A and B respectively. Similarly, σ_A and σ_B represent the standard deviation of image A and B respectively. In general, for any plain-text image there exists high correlation between adjacent pixels, whereas in the ciphered image it should be close to 0.

4.1.3 *NPCR* and *UACI* measures

The number of changing pixel rate ($NPCR$) and unified averaged changed intensity ($UACI$) of an image are the commonly used parameters to measure the change in encrypted pixels by modifying the value of a single-pixel in the original image. These metrics are commonly used to evaluate the strength of ciphers for differential attacks. For a 256-gray level image, the NPCR and UACI between two images are defined as

$$NPCR_{A,B} = \frac{\sum_{i=1}^{N_L} \sum_{j=1}^{N_C} D(i, j)}{N_L \times N_C} \times 100 \tag{29}$$

where

$$D(i,\, j) = \begin{cases} 1, & \text{if } A(i,\, j) \neq B(i,\, j); \\ 0, & \text{otherwise.} \end{cases} \tag{30}$$

and

$$UACI_{A,B} = \frac{100}{255} \frac{\sum_{i=1}^{N_L} \sum_{j=1}^{N_C} |A(i,\, j) - B(i,\, j)|}{N_L \times N_C} \tag{31}$$

where $\mathscr{F} = 255$ is the largest supported value of 256 gray level images. A high NPCR ($NPCR > 99.5810$) and UACI ($33.3445 \leq UACI \leq 33.5826$) [21] imply a high resistance of the cipher to differential attacks.

4.2 Key-space analysis

The key-space analysis is performed by evaluating the key-space and key sensitivity.

4.2.1 Key-space

Key-space is an ensemble of all possible combinations of keys that are used for encryption. A 256-bit key is known to be sufficiently large to prevent brute force attacks. One of the proposed cipher's main advantages is its key-space's extensibility. To simulate a post-quantum computing attack, we extended the key to 2048 bits. Indeed, most chaos-based ciphers exhibit a large key-space with keys that cannot easily be proven to be different. However, the sensitivity of the key is verified by changing its least significant bit (LSB). However, by our approach, the key-space can be extended as desired without sacrificing the independence of the keys. This approach consists of affecting to each sub-image j a sub-key ($\mathbf{a}(k)$, $\mathbf{b}(k)$, $\mathbf{c}(k)$, $\mathbf{d}(k)$, $\mathbf{e}(k)$, $\mathbf{f}(k)$, $\mathbf{g}(k)$, $\mathbf{h}(k)$) corresponding to a map QACM_k, with $k = 1 + j \bmod \kappa$. Each sub-image j is encrypted with a different map QACM_k, such that any permutation in the set $\{\text{QACM}_k\}_{1 \leq k \leq \kappa}$ affects the behaviour of the cryptogram, thus giving the possibility to extend the key-space.

4.2.2 Sensitivity of the key

Key sensitivity measures the sensitivity of the encryption algorithm to a small change in the key value. A high sensitivity of the key is required to prevent adaptive chosen-plaintext attacks and linear cryptanalysis. To evaluate the sensitivity of our cipher to the external key, we have encrypted the same image with two slightly different keys $K1$ as mentioned above and $K2 = azertyuiopqsdfghazertyuiopqsdfg1$. The key $K2$ has only one-bit change from key $K1$. The plain-text image is encrypted with keys $K1$ and $K2$; UACI, NPCR, and correlation coefficients between the two encrypted images are tabulated in Table 4. The NPCR and UACI values are more than the reference values, and correlation coefficients close to zero suggest that the algorithm is highly sensitive to the key. In addition, the entropy of the image encrypted with key $K1$ and decrypted with key $K2$ is computed and tabulated in the same table. The entropy close to eight demonstrates that the decryption is unsuccessful; hence, the proposed algorithm is extremely sensitive to the key. We repeated the same experiment with a 2048-bit key. We set the 2048-bit as $K'1 = K1K1K1K1K1K1K1K1$ and a one-bit different key $K'2$ as $K'2 = K1K1K1K1K1K1K1K2$. The sensitivity of the key was evaluated under the same conditions as for the 256-bit key and the results are tabulated in Table 4. It demonstrates that the sensitivity of the key analysis is the same as in the case of the 256-bit key. Therefore, we can conclude that the proposed cipher presents an extensible key space than can be adapted depending on the desired security level.

Table 4 Detailed statistical properties of images encrypted with two slightly different keys $K1$ and $K2$ in the case of 256-bit, and $K'1$ and $K'2$ in the case of 2048-bit

		256-bit				2048-bit			
Image	Color	NPCR	UACI	Entropy	Correlation	NPCR	UACI	Entropy	Correlation
Lena	Red	99.6078	33.4238	7.9994	0.0017	99.6239	33.4431	7.9994	0.0004
	Green	99.6098	33.4896	7.9994	− 0.0015	99.5922	33.4277	7.9991	0.0000
	Blue	99.6136	33.5038	7.9993	− 0.0010	99.6296	33.5368	7.9993	− 0.0043
Baboon	Red	99.6048	33.4987	7.9994	0.0012	99.6002	33.4614	7.9994	0.0004
	Green	99.5956	33.4434	7.9992	0.0006	99.6178	33.5194	7.9992	− 0.0010
	Blue	99.6243	33.4855	7.9992	− 0.0015	99.6094	33.4075	7.9993	0.0017
Airplane	Red	99.6063	33.4631	7.9993	0.0007	99.5899	33.4199	7.9992	0.0004
	Green	99.6208	33.4720	7.9993	− 0.0014	99.6155	33.4583	7.9992	0.0008
	Blue	99.5884	33.5359	7.9993	− 0.0043	99.6162	33.4986	7.9994	0.0005
Peppers	Red	99.6117	33.5103	7.9993	− 0.0017	99.6014	33.5019	7.9993	− 0.0027
	Green	99.6426	33.3958	7.9992	0.0028	99.5899	33.5445	7.9993	− 0.0033
	Blue	99.6315	33.3403	7.9988	0.0039	99.6273	33.4066	7.9993	0.0006

$NPCR$, $UACI$ and entropy are to be compared with reference values $NPCR > 99.5810$ and $33.3445 \leq UACI \leq 33.5826$ and 8 respectively. For this experiment, we used $R = 3$ rounds

In Fig. 4, an example of ciphering/deciphering realized with two ($R = 2$) rounds is presented. The plain-image is encrypted with $K'1$. After that, the ciphered image is successfully decrypted with $K'1$. However, the decryption fails with $K'2$, which attests to the high sensitivity of the cipher to the encryption key.

Security of the proposed cipher, although we set $T_1 = T_2 = 2$. Figure 5 shows the behavior of the entropy values of the RED component of the image of Lena encrypted with $K1$ and those of the corresponding attempt for decryption using $K2$. The entropy is evaluated by varying T_1 and T_2 and plotted in terms of $\mu_1 = \log_2(T_1)$ and $\mu_2 = \log_2(T_2)$ in Fig. 5.

From this figure, we observed that the sensitivity of the key depends on the choice of (T_1, T_2). We verified that the system is secure to one-bit change in the external key as $(\mu_1, \mu_2) < (4, 4)$. The upper limit $\mu_m = 4$ of μ_1 and μ_2 corresponds to the binary logarithm of the block length of the confusion-diffusion step. We observe that the entropy decreases with an increase in T_1 or T_2, leading to a decrease in the key sensitivity. Further, in some cases, entropy values are close to those of the plain-image entropy ($H_R = 7.2531$). It attests to the vulnerability of the proposed scheme for these values of μ_1 and μ_2. In the proposed method, such a sensitivity decrease is compensated by increasing the number of rounds of the algorithm.

4.3 Statistical analysis

The histogram, the correlation of adjacent pixels (ρ_h : correlation coefficient between horizontal adjacent pixels; ρ_v : correlation coefficient between vertical adjacent pixels; ρ_d : correlation coefficient between diagonal adjacent pixels) and the information entropy of the ciphered image are evaluated for several 256 gray-scale images. Figure 6 shows the results

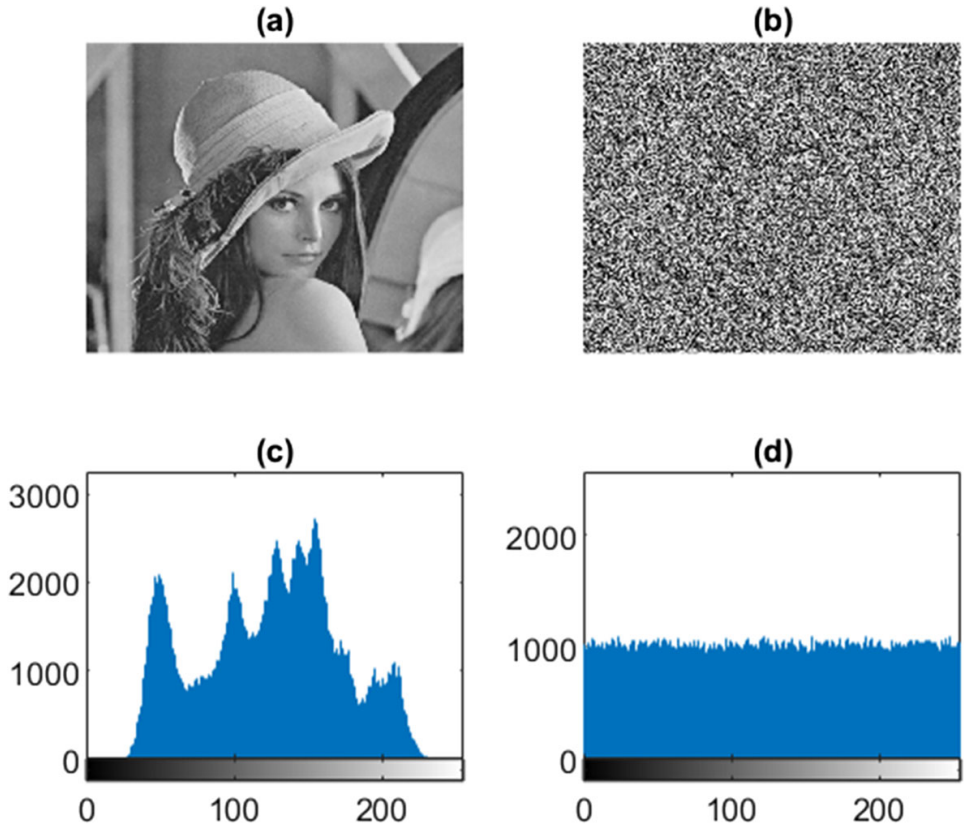


Fig. 4 Sensitivity of the key to one-bit change: (a) successfully decrypted (original) image with $K'1$, (b) unsuccessfully decrypted image with $K'2$, (c) histogram of the original image and (d) histogram of the unsuccessfully decrypted image

obtained for the color image of Lena (Fig. 4(a)) by varying number of rounds R between 1 and 10. Figure 6 suggests that the entropy is independent of the number of rounds R for $R > 1$, thereby attests that the minimum number of rounds required for the cipher to be secure is $R = 2$. Similarly, it also shows a satisfactory analysis result for the correlation of horizontally (ρ_h), vertically (ρ_v), and diagonally (ρ_d) adjacent pixels. Indeed, it can be concluded that the correlation coefficients of the three image components, i.e., RED, GREEN, and BLUE, are close to zero, independently of the number of rounds R. Thus, the proposed cipher satisfies the zero-correlation property necessary to resist statistical attacks.

Figure 7 shows the histograms of two round ciphered images of Lena for the RED, GREEN, and BLUE components. It appears that the histogram of each encrypted component is fairly uniform and significantly different from that of the corresponding plain-image component. It demonstrates that deducing the secret key from the ciphertext during the known/chosen plaintext attacks is hard.

4.4 Differential attack

For the cipher to resist differential attacks, it must be sensitive to a small change (single-pixel change) in the plain-image. We evaluated the robustness of our cipher against the differential attacks by comparing the $NPCR$ and $UACI$ values of a two-round ciphered image of Lena to their reference values. We diagonally varied the position (x, y) of changed pixels as (k, k), with $1 \leq k \leq 512$ by the step size of 2, and computed the corresponding $NPCR$ and $UACI$ for the RED, GREEN and BLUE components of the color image of Lena, and plotted in Fig. 8.

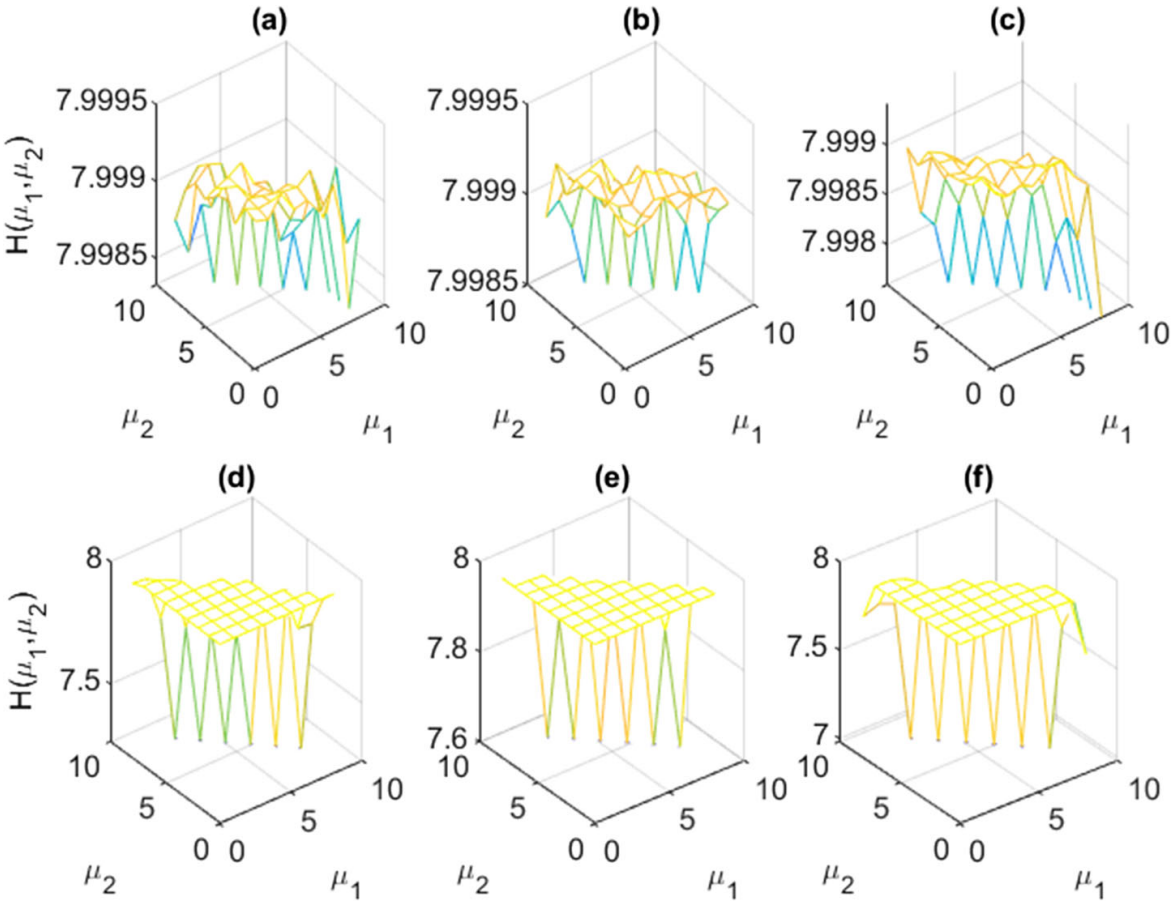


Fig. 5 Behaviour in term of μ_1 and μ_2 of entropy values of the ciphered and decrypted components of the color image of Lena($R = 2$): (a)-(c) cases of the respective RED, GREEN and BLUE components encrypted with $K1$; and (d)-(f) cases of attempt for decryption with $K2$ for RED, GREEN and BLUE components, respectively. Satisfactory entropy values after an attempt for decryption are observed for $(\mu_1, \mu_2) < (4, 4)$

A cipher is secure as $NPCR > 99.5810$ and $33.3445 \leq UACI \leq 33.5826$ ($\nu = 0.01$ significance level) for gray images of size 512×512 [21]. The result in Fig. 8 shows that our cipher is sensitive to one-pixel change for $R > 1$. From this figure, it appears that the proposed cipher can resist differential attacks as the $NPCR$ and $UACI$ remain in the good range independently of the coordinate of the pixel change. We also observed that the values of the $NPCR$ and $UACI$ depend on the input image. Indeed, for two different components, the $NPCR$ values obtained from the same pixel change coordinate are different; the same observation is made for the $UACI$. Table 5 shows comparative values of the proposed cipher's $NPCR$ and $UACI$ and other recent cipher ($R = 3$). This comparison also confirms the effectiveness of our algorithm.

4.5 Speed performance analysis

The execution speed of the proposed cipher is evaluated using Matlab 18b in the CPU as specified above. Table 6 compares the execution speed of the proposed algorithm with other chaos-based ciphers. We used the gray-scale images of cameraman (256×256) and Lena (512×512) for this experiment. The average execution time, for $R = 2$, is about $0.4478\,s$ for the 512×512 gray-scale images. Although this speed is comparable to that of Refs. [1] and [42], it remains low as compared to the one obtained in [13]. Such a low execution speed is justified by the multiple data conversion involved in the algorithm, which is not necessary

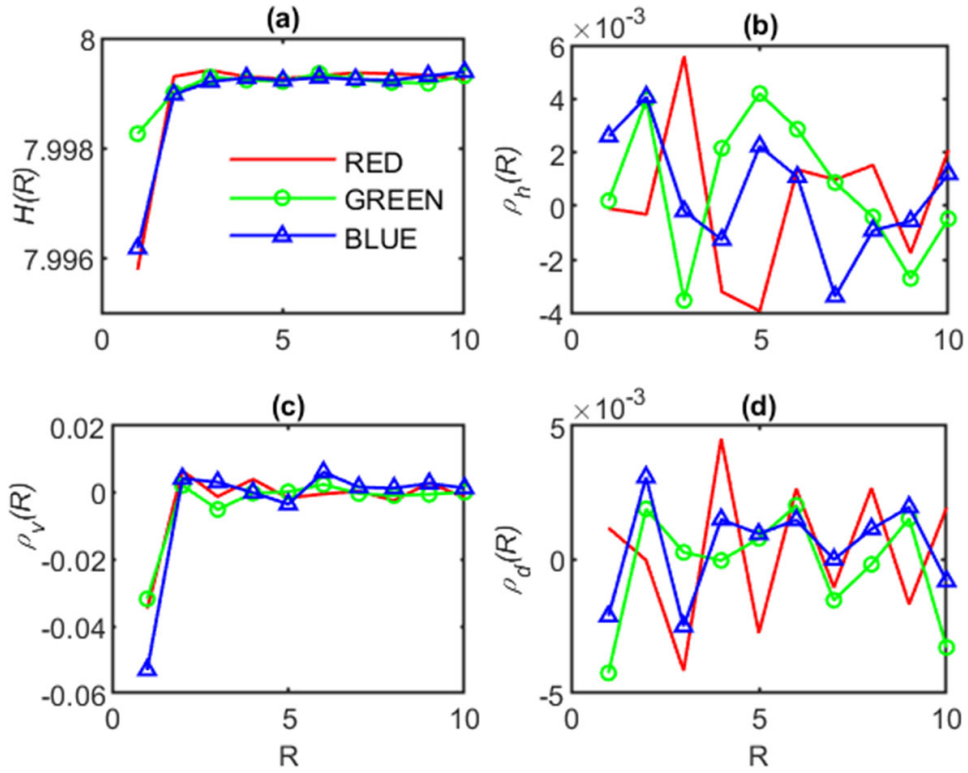


Fig. 6 Entropy values H and correlation coefficients ρ in terms of the number of rounds R. (a) Entropy values (H), (b)-(d) correlation coefficients of respectively horizontal (ρ_h), vertical (ρ_v) and diagonal (ρ_d) adjacent pixels for the RED, GREEN and BLUE components

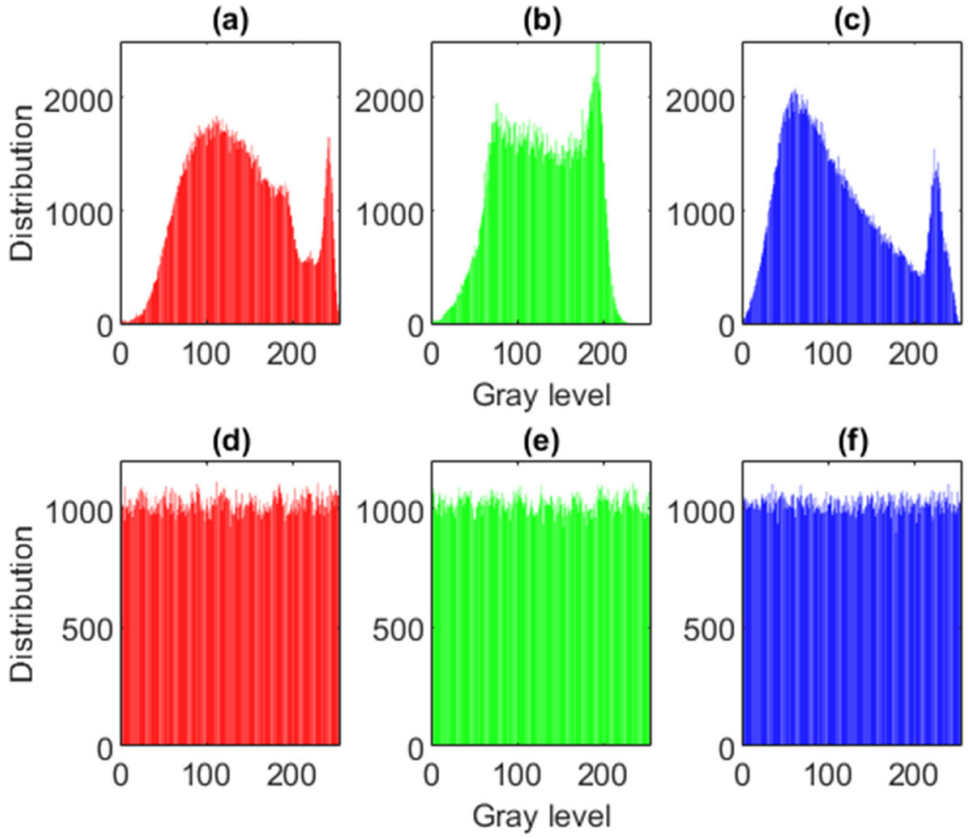


Fig. 7 Histograms of the image of Lena: first line shows from left to right histograms of the (a) RED, (b) GREEN and (c) BLUE components of the original image, respectively; second line (d)-(f) shows histograms of the corresponding components of the ciphered image

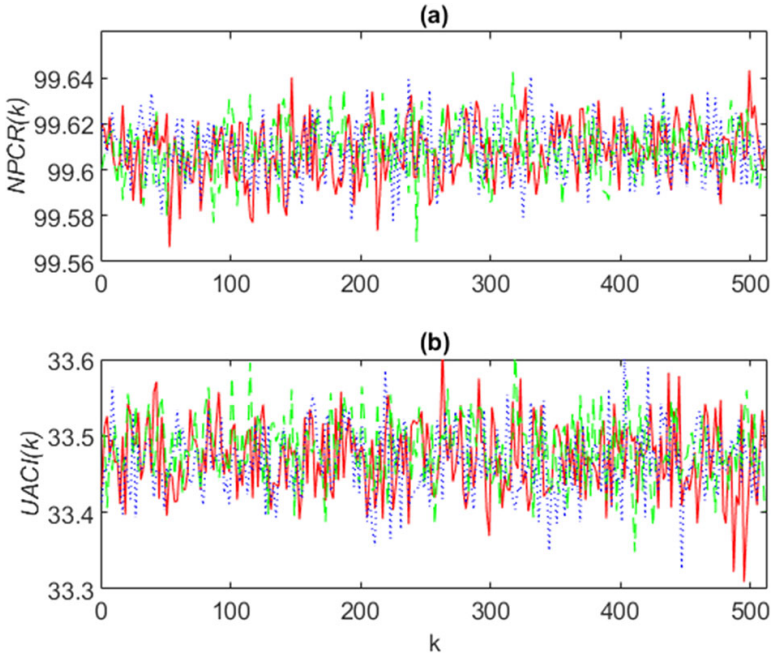


Fig. 8 Dependence of NPCR and UACI on position $((k, k))$ of the changed pixel: case of the color image of Lena. The change applies to each image component and the NPCR of the RED, GREEN and BLUE components are computed separately for each of the components

for hardware implementation. For example, the decomposition of the pixel gray-level into two 4-bit encoded integers $q(t)$ and $r(t)$ implies a division and a modulo operation. In the hardware implementation, these operations correspond to the four most significant bits (MSB) as the quotient $q(t)$ and the four least significant bits (LSB) as the remainder $r(t)$, thereby is time-saving. Similarly, the inverse of this decomposition is time costly in Matlab and corresponds to the concatenation of the two 4-bit outputs $q(t + 1)$ and $r(t + 1)$ in the hardware implementation. Thus, the proposed algorithm is more suitable for low-cost hardware implementation than software implementation. The overall comparison shows that the cipher in Ref. [13] is running faster in the Matlab environment than in ours; however, it requires floating-point arithmetic that is hardware costly than the 4-bit integer arithmetic used in the proposed scheme. Our algorithm offers the advantage of combining only 4-bit and 8-bit integer arithmetic operations, precisely addition and subtraction. These basic operations make simpler its hardware implementation even with low-end microprocessors

Table 5 Comparison of $NPCR$ and $UACI$ values for color images

		$NPCR$ (%)			$UACI$ (%)		
Cipher	Image	RED	GREEN	BLUE	RED	GREEN	BLUE
Proposed	Baboon	99.6429	99.6437	99.6464	33.5764	33.5881	33.6311
	Peppers	99.6479	99.6410	99.6403	33.6121	33.5629	33.5863
Ref. [32]	Baboon	99.6246	99.5914	99.5972	33.4702	33.4641	33.4625
	Peppers	99.6315	99.6017	99.6380	33.5577	32.7183	33.5702
Ref. [40]	Baboon	99.6056	99.6164	99.6256	33.4478	33.4495	33.4401
	Peppers	99.6098	99.6218	99.5948	33.5012	33.4415	33.4536

Table 6 Comparison of encryption time of different algorithms

Image size	Execution time (s)			
	Proposed cipher	Ref. [13]	Ref. [1]	Ref. [42]
256×256	0.1335	0.0202	0.1789	0.1950
512×512	0.4478	0.0708	0.6639	0.6500

The average execution time (in second) of the proposed cipher are obtained with $R = 2$ and $K1$ as encryption key

without losing its security properties. Moreover, the algorithm can easily be parallelized according to its architecture shown in Fig. 2.

5 Conclusion

This paper proposed an extendable integer image cipher based on the combined 4-bit and 8-bit PWLCM. The cipher does not require any multiplication operation. The PWLCM is obtained by perturbing the convectional QACM and presents an extended period. The key space extensibility is achieved using different lattice length (κ) of the PWLCM control vectors. The extended key space helps to avoid key duplication. We evaluated the sensitivity of the key under 256-bit and 2048-bit key conditions and verified the high sensitivity of the key for both cases. Such flexibility for the extensibility of the key-space makes the proposed cipher a good candidate to resist brute-force attacks even under the post-quantum computing situation. The main advantages of the proposed cipher are the low number of bits involved in its hardware implementation and the extensibility of its key-space. Moreover, only unsigned integer addition and subtraction operations are used, contrary to other chaos-based ciphers that involve floating-point arithmetics, including time-consuming operations like multiplications and divisions. The statistical, differential, and key-space analysis demonstrate the robustness of the proposed cipher against known attacks. We further expect to reduce the block size while maintaining a high-security level of the cipher, simplifying its implementation with low-end processors under the limited memory space constraints.

Acknowledgements This work was partially supported by the Erasmus+ program under Ref KA107 Call 2019.

We thank the reviewers for their valuable comments.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data Availability Data sharing not applicable to this article as no datasheets were generated or analyzed during the current study.

Declarations

Conflict of Interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Ahmed A, El-Latif A, Li L, Niu X (2014) A new image encryption scheme based on cyclic elliptic curve and chaotic system. Multimed Tools Appl 70:1559–1584
2. Atawneh S, Almomani A, Bazar H, Sumari P, Gupta BB (2017) Secure and imperceptible digital image steganographic algorithm based on diamond encoding in DWT domain. Multimed Tools Appl 76:18,451–18,472
3. Bajard JC, Eynard J, Merkiche N (2018) Montgomery reduction within the context of residue number system arithmetic. J Cryptogr Eng 8:189–200
4. Chen F, Wong KW, Liao X, Xiang T (2012) Period distribution of generalized discrete Arnold cat map for $n = p^e$. IEEE Trans Inf Theory 58:445–452
5. Chui CK, Chen G, Mao Y (2004) A symmetric image encryption scheme based on 3D chaotic cat maps. Chaos, Solitons Fractals 21:749–761
6. Crutchfield JP (1998) Spatio-temporal complexity in non linear image processing. IEEE Trans Circ Syst 35:770–780
7. Cui M, Tong X (2008) Image encryption with compound chaotic sequence cipher shifting dynamically. Image Vis Comput 26:843–850
8. Didier LS, Dosso FY, Véron P (2020) Efficient modular operations using the adapted modular number system. J Cryptogr Eng 10:111–133
9. Djeugoue H, Gnyamsi GG, Eyebe Fouda JSA, Koepf W (2022) On the implementation of large period piece-wise linear Arnold cat map. Multimed Tools Appl. https://doi.org/10.1007/s11042-022-13175-6
10. Dyson FF, Falk H (1992) Period of a discrete cat mapping. Am Math Mon 99:603–614
11. Elshamy AM, Hussein AI, Hamed HFA, Abdelghany MA, Kelash HM (2019) Color image encryption technique based on chaos. Procedia Comput Sci 163:49–53
12. Eyebe Fouda JSA, Koepf W (2022) An 8-bit precision cipher for fast image encryption. Multimed Tools Appl. https://doi.org/10.1007/s11042-022-12368-3
13. Eyebe Fouda JSA, Effa JY, Ali M (2014) A fast chaotic block cipher for image encryption. Commun Nonlinear Sci Numer Simulat 19:578–588
14. Eyebe Fouda JSA, Effa JY, Ali M (2014) Highly secured chaotic block cipher for fast image encryption. Appl Soft Comput 25:435–444
15. Fridrich J (1998) Symmetric ciphers based on two-dimensional chaotic maps. Int J Bifurcat Chaos 8:1259–1284
16. Gu G, Linga J (2014) A fast image encryption method by using chaotic 3d cat maps. Optik 125:4700–4705
17. Herbert V, Biswas B, Fontaine C (2019) Design and implementation of low-depth pairing-based homomorphic encryption scheme. J Cryptogr Eng 9:185–201
18. Hua Z, Zhou Y, Huang H (2019) Cosine-transform-based chaotic system for image encryption. Inf Sci 480:403–419
19. Ilan Y (2019) Generating randomness: making the most out of disordering a false order into a real one. J Transl Med 17(49):1479–5876
20. Kaixin J, Ye G, Dong Y, Huang X, He J (2020) Image Encryption Scheme based on Generalized Arnold Map and RSA Algorithm. Secur Commun Netw 2020:9721,675
21. Kang S, Liang Y, Wang Y, VI M (2019) Color image encryption method based on 2D-variational mode decomposition. Multimed Tools Appl 78:17,719–17,738
22. Kaur G, Agarwal R, Patidar V (2020) Chaos based multiple order optical transform for 2d image encryption. Eng Sci Technol Int J 23:998–1014
23. Keating JP, Mezzadri F (2000) Pseudo-symmetries of Anosov map and spectral statistics. Nonlinearity 13:747–775

24. Kopparthi VR, Kali A, Sabat SL, Anumandla KK, Rangababu P, Eyebe Fouda JSA (2022) Hardware architecture of a digital piecewise linear chaotic map with perturbation for pseudorandom number generation. Int J Electron Commun (AEÜ) 147:154,138
25. Li D, Deng L, Gupta BB, Wang H, Choi C (2019) A novel cnn based security guaranteed image watermarking generation scenario for smart city applications. Inf Sci 479:432–447
26. Lian S, Sun J, Wang Z (2005) A block cipher based on a suitable use of the chaotic standard map. Chaos, Solitons Fractals 26:117–129
27. Liao X, Xiang T, Wong KW (2007) Selective image encryption using a spatio temporal chaotic system. Chaos 17:0231,151–0231,512
28. Malina L, Popelova L, Dzurenda P, Hajny J, Martinasek Z (2018) On feasibility of post-quantum cryptography on small devices. IFAC PapersOnLine 51-6:462–467
29. Mondal B, Behera PK, Gangopadhyay S (2021) A secure image encryption scheme based on a novel 2D sine–cosine cross-chaotic (SC3) map. J Real-Time Image Proc 18:1–18
30. Panwar K, Purwar R, Jain A (2018) Cryptanalysis and improvement of an image encryption scheme using combination of one-dimensional chaotic maps. J Electron Imaging 27:053,037
31. Pareek NK, Patidar V, Sud KK (2006) Image encryption using chaotic logistic map. Image Vis Comput 24:926–934
32. Patro KAK, Acharya B (2019) An efficient colour image encryption scheme based on 1-D chaotic maps. J Inf Secur Appl 46:23–41
33. Ping P, Fan J, Mao Y, Xu F, Gao J (2018) A chaos based image encryption sheme using digit-level permutation and block diffusion. IEEE Access 6:67,581–67,593
34. Ping P, Xu F, Mao Y, Wang Z (2018) Designing permutation-substitution image encryption networks with Henon map. Neurocomputing 283:53–63
35. Schoinianakis D (2020) Residue arithmetic systems in cryptography: a survey on modern security applications. J Cryptogr Eng 10:249–267
36. Wang M, Wang X, Zhang Y, Zhou S, Zhao T, Yao N (2019) A novel chaotic system and its application in a color image cryptosystem. Opt Lasers Eng 121:479–494
37. Wikramaratna RS (2008) The additive congruential random number generator—a special case of a multiple recursive generator. J Comput Appl Math 216:371–387
38. Wu Y (2012) Image encryption using the two-dimensional logistic chaotic map. J Electron Imag 21:013,014
39. Ye G, Huang X (2016) A secure image encryption algorithm based on chaoticmaps and SHA-3. Secur Comm Netw 9:2015–2023
40. Zhang Y, He Y, Li P, Wang X (2020) A new color image encryption scheme based on 2DNLCML system and genetic operations. Opt Lasers Eng 128:106,040
41. Zhao Y, Gao C, Liu J, Dong S (2019) A self-perturbed pseudo-random sequence generator based on hyperchaos. Chaos, Soliton Fractals: X 4:100,023
42. Zhu ZL, Zhang W, Wong KW, Yu H (2011) A chaos-based symmetric image encryption scheme using a bit-level permutation. Inf Sci 181:1171–1186
43. Zhua H, Zhao C, Zhanga X, Yanga L (2014) An image encryption scheme using generalized Arnold map and affine cipher. Optik 125:6672–6677

Affiliations

Gaetan Gildas Gnyamsi Nkuigwa[1] · Hermann Djeugoue Nzeuga[1] · J. S. Armand Eyebe Fouda[1,2] · Samrat L. Sabat[3] · Wolfram Koepf[2]

Gaetan Gildas Gnyamsi Nkuigwa
gaetangildas@yahoo.fr

Hermann Djeugoue Nzeuga
mahernzeuga@yahoo.fr

Samrat L. Sabat
slssp@uohyd.ac.in

Wolfram Koepf
koepf@mathematik.uni-kassel.de

1 Départment de Physique, Université de Yaoundé I, Yaoundé, Cameroun

2 Institute of Mathematics, University of Kassel, Kassel, Germany

3 Centre for Advanced Studies in Electronics Science and Technology, University of Hyderabad, Hyderabad, India