# Théorie du contrôle et systèmes hybrides dans un contexte cryptographique

Phuoc Vo Tan

Nancy-Université INPL

**École doctorale IAEM Lorraine**
**DFD Automatique et Production Automatisée**
**Institut National Polytechnique de Lorraine**

# Théorie du contrôle et systèmes hybrides dans un contexte cryptographique

# THÈSE

présentée et soutenue publiquement le 12/11/2009

pour l'obtention du

**Doctorat de l'Institut National Polytechnique de Lorraine**

**Spécialité Automatique et Traitement du signal**

par

Phuoc VO TAN

**Composition du jury**

| | | |
|---|---|---|
| *Président :* | Mohammed MSAAD | Professeur<br>Ensicaen,Caen |
| *Rapporteurs :* | Nacim RAMDANI | Professeur<br>Université Montpellier 2, Montpellier |
| | Krishna BUSAWON | Professeur<br>Northumbria University, Royaume-Uni |
| *Examinateur :* | Philippe GUILLOT | Maître de Conférences,<br>Université Paris 8, Paris |
| *Directeur de thèse :* | Gilles MILLERIOUX | Professeur<br>Université Henri Poincaré - Nancy 1, Nancy |
| *Co-directeur de thèse :* | Jamal DAAFOUZ | Professeur<br>Institut National Polytechnique de Lorraine, Nancy |

CRAN

**Centre de Recherche en Automatique de Nancy**
**UMR 7039 Nancy-Université – CNRS**
**2, avenue de la forêt de Haye 54516 Vandœuvre-lès-Nancy**
**Tél. +33 (0)3 83 59 59 59 Fax +33 (0)3 83 59 56 44**

Mis en page avec la classe thloria.

# Remerciements

Avant d'aborder le thème de cette thèse, je tiens à remercier tous ceux qui, de près ou de loin, m'ont permis d'effectuer ces travaux.

C'est à Gilles Millérioux et Jamal Daafouz, mon directeur et co-directeur de thèse, que j'exprime en premier lieu ma profonde reconnaissance, eux qui depuis tant d'années m'ont fait progresser en restant toujours respectueux de ma personne. Ils m'ont fait bénéficier non seulement de leurs connaissances scientifiques mais aussi de leurs méthodes de travail.

Je souhaite adresser mes remerciements au professeur Claude Iung, à Mme Christine Pierson, à mon tuteur pédagogique Gérard Bloch pour m'avoir accueilli et m'avoir conseillé depuis mon premier jour au CRAN.

Je désire remercier le jury qui a accepté d'évaluer cette thèse.

Je remercie l'ensemble du personnel du CRAN et de l'ESSTIN en particulier, de l'AIP Lorraine qui ont contribué à créer une ambiance de travail sympathique : Luc Lossent, Yves Lordonne, Phillipe Reb. Je retiens particulièrement les cafés sympas de Jean-Luc Noisette et Eric Gnaedinger. Merci à Sylvie Ferrari pour sa disponibilité et sa serviabilité.

Je remercie M. Jean-Marie Ory et M. Jérémy Parriaux pour m'avoir guidé concernant les aspects FPGA et DSP.

Je voudrais également exprimer mes sincères remerciements à mes amis, Floriane Anstett Collin et Fabien Lauer. Ils m'ont permis de découvrir un autre univers de Linux, de Latex, de l'automatique et de la culture française.

Enfin, je dédie cette thèse à mes parents, à ma tante, à mes deux frères, à mon amour.

# Table of contents

# Notations

| | |
|---|---|
| SISO | Single Input Single Output |
| MIMO | Multi Input Multi Output |
| SSC | Synchronous Stream Cipher |
| SSSC | Self-Synchronizing Stream Cipher |
| CFB | Cipher Feedback Mode of operation |
| | |
| $\mathbb{R}$ | Set of real numbers |
| $\mathbb{N}$ | Set of natural numbers |
| $\mathbb{N}^+$ | Set of positive natural numbers |
| $\mathbb{Z}$ | Set of integer numbers |
| $\mathbb{Z}_p$ | Set of remainders in arithmetic modulo $p$ |
| $\mathbb{F}_p$ | Set of remainders in arithmetic modulo $p$ when $p$ is prime (finite field) |
| $U_p$ | subset of $\mathbb{F}_p$ |
| $\mathbb{F}_p^n$ | Set of vectors of dimension $n$ whose components are in $\mathbb{F}_p$ |
| $\mathbb{F}_p[z_k]$ | Ring of polynomials whose indeterminates $z_k^{(i)}$ and the coefficients are in $\mathbb{F}_p$ |
| $I, U, V$ | Dense or compact sets |
| $l$ | Distance in a topological space |
| | |
| $f$ | State transition function |
| $h$ | Output function |
| $f^{(i)}$ | $i$-order iterated state transition function |
| $h^{(i)}$ | $i$-order iterated output function |
| $m_k$ | Inputs of a dynamical system |
| $y_k$ | Outputs of a dynamical system |
| $x_k$ | Internal state of a dynamical system |
| $x_0$ | Initial condition |
| | |
| $A$ | Set of inputs $m_k$ |
| $B$ | Set of outputs $y_k$ |
| $X$ | Set of internal states $x_k$ |
| | |
| $n$ | Dimension of a dynamical system |
| $m$ | Number of inputs |
| $p$ | Number of output |

| | |
|---|---|
| $\theta$ | Parameters of dynamical system |
| $L$ | Dimension of the parameter vector $\theta$ |
| | |
| $r$ | Left inherent delay of dynamical system |
| $R$ | Relative delay of dynamical system |
| $K$ | Flatness characteristic number of a dynamical system |
| | |
| $\tilde{f}, \tilde{h}$ | Functions describing the receiver part of a cryptosystem |
| $\hat{m}_k$ | Reconstructed input at the receiver part |
| $\hat{y}_k$ | Reconstructed output at the receiver part |
| $\hat{x}_k$ | State vector at the receiver part |
| $r_k^i$ | Residual for detection at the receiver part |
| | |
| $\sigma$ | Switching function |
| $J$ | Number of modes of the switching function $\sigma$ |
| $N$ | Number of input/output relations or numbers of receivers in a bank |
| $M_N(K)$ | Number of monomials of a Veronese map |
| | |
| $z_k$ | Regressor vectors in an identification procedure |
| $N'_{max}$ | Maximum number of regressor vectors |
| $b_t$ | Parameter vector in an identification procedure |
| $p_N(z_k)$ | Hybrid decoupling constraint polynomial |
| $h_N$ | Coefficient of hybrid decoupling constraint polynomial |
| $N_{I/O}$ | Maximum number of input/output relations |
| $\mathcal{L}_N$ | Matrix of regressor vectors |
| $l_{i,j}^{[k]}$ | $k+1$ order minor of a matrix $l_{ij}$ |
| | |
| $x^*$ | Stationnary state |
| $K^o, K'^o$ | Order of a periodic orbit |
| $T$ | Particular discrete time $k$ |
| | |
| $\{\ \}$ | Family of elements |
| $\{a\}_{k_1}^{k_2}$ | Sequence of symbols $a_k$ between $k = k_1$ and $k = k_2$ |
| $\mathcal{T}$ | Compound matrix in the Extended Euclidean Algorithm for computing the multiplicative inverse |
| $gcd(a, b)$ | Greatest common divisor between $a$ and $b$ |
| | |
| $V$ | Lyapunov function |
| $P$ | Positive definite matrix of a Lyapunov function |

| | |
|---|---|
| $\mathbf{1}_n$ | $n$ dimensional identity matrix |
| $\mathbf{0}$ | Zero matrix with appropriate dimension if not specified |
| $\|x_k\|$ | Euclidean norm of vector $x_k$. $\|x_k\| = \sqrt{x_k^T x_k}$ |
| $X^\dagger$ | Moore-Penrose pseudo inverse of $X$ |
| $X^T$ | Transpose of the matrix $X$ |
| $Ker(X)$ | Kernel (null space) of $X$ |
| $eig(X)$ | Eigevalues of $X$ |
| $rank(X)$ | Rank of $X$ |
| | |
| $\nu_e$ | Encoding function in a two-channel transmission |
| $\nu_d$ | Decoding function in a two-channel transmission |
| $e$ | Encryption function in stream ciphers |
| $d$ | Decryption function in stream ciphers |
| $\sigma^s$ | Keystream generation function in SSC |
| $s$ | Ouput function in SSC |
| $\sigma^{ss}$ | Keystream generation function in SSSC |
| $q_k$ | Internal vector of automata in SSSC based on Maurer approach |
| $g_\theta$ | Next state transition function in automata based on Maurer approach |
| $M$ | Delay of memorization in SSSC |
| $h'$ | Filter function of an SSSC in CFB mode |
| $l_\theta$ | Function describing in terms of past ciphertexts |
| | |
| $e_j^k$ | Expansion rate of trajectories in an attractor |
| $\lambda_L$ | Lyapunov Exponent |
| $\lambda$ | Divergence rate of two trajectories or eigenvalues |

# Introduction

Hybrid systems have inspired a great deal of research from both control theory and theoretical computer science. They provide a strong theoretical foundation which combines discrete-event and continuous-time systems in a manner that can capture software logic and physical dynamics in a unified modelling framework. The most well-known area of applicability of hybrid systems are naturally modelling, analysis and control design of embedded systems. Switched systems are an important class of hybrid systems that are widely studied in the literature. Stability, identifiability, controller or observer design are challenging problems related to hybrid system. They are usually addressed for engineering applications. This manuscript deals with a specific engineering application : secure communications. As it will turn out, dynamical systems will play a central role in this context.

Chaotic behavior is one of the most complex dynamics a nonlinear system can exhibit. One of the formal definitions of chaos is due to R.L. Devaney [Dev89]. A dynamical system is said to be chaotic in the sense of Devaney if it fulfills two properties : transitivity and density of periodic points. It can be shown that sensitivity to the initial condition, which is the property most often associated with chaotic behavior, is actually a consequence of those two others properties. Roughly speaking, a system is said to be sensitive to initial conditions if a small change in the initial condition drastically changes the behavior of a system in the long run, thus making long-term predictions unfeasible in practice. Complex dynamics had its beginnings in the work of the French mathematician Henri Poincaré (1854-1912), who also recognized the practical unpredictability of such systems. Sensitive dependent phenomena were highlighted by Edward Lorenz in 1963 while simulating a simplified model of convection. But it was the paper "Period Three Implies Chaos" by Li and Yorke in 1975 [LY75], where the word "chaos" was coined in the framework of dynamical systems, which triggered a tremendous interest in this kind of phenomena.

Signals generated by chaotic systems are broadband, noiselike and present random-like statistical properties, in spite of being deterministic. This makes them a very convenient tool to implement the principles of confusion and diffusion required by Shannon in cryptography [Sha49][Mas92]. The first ideas in this direction were made around 1990 [Mat89][HNSM91]. Since the 90's, many schemes, also called cryptosystems or ciphers, have been proposed to scramble information with chao-

tic signals. The papers [CP91] or [MVH93] can be considered as pioneering works on this topic. This new approach to encryption is commonly called chaos-based (or "chaotic") cryptography. To illustrate the high research activity in this field, let us mention than many special issues have been already published in international journals like IEEE Transactions on Circuits and Systems, or International Journal on Bifurcations and Chaos ([Ogo93][Has98][Yan04][AL06][MAD08] are some important surveys dealing with the topics). Furthermore, numerous invited sessions on chaos-based cryptography have been organized at international conferences, e.g., the International Symposium on Circuits And Systems (ISCAS), the International Symposium on NOnLinear Theory and Applications (NOLTA), and many others.

This being the case, two points deserve important comments and further consideration. First, chaos-based cryptographic primitives were most often considered as secure exclusively because of the complexity of the dynamics which is exhibited. However, observe that when chaotic generators are implemented on machines with finite accuracy (say, a computer), the sequences are not really chaotic. Indeed, since the variables take values in sets of finite cardinality, such sequences obviously get trapped in a loop, called *cycle*, of finite period. We can expect this period to be not too short and the degree of "randomness" of the sequence to be high (as measured e.g. by standard statistical tests), but guaranteeing the said properties requires some caution [Knu98]. Important contributions to this issue and a definition of so-called *discrete chaos* can be found in [KSAT06]. Secondly, not enough attention has been paid on the basic rules borrowed from standard cryptography an encryption scheme should obey, in particular the fundamental assumption first stated by A. Kerkhoff in ([DK02]). This assumption states that any unauthorized person (called adversary or eavesdropper) knows all the details of the cryptosystem, including the algorithm and its implementation, except the secret key. More generally, the cryptanalysis, that is the study of attacks against cryptographic schemes in order to reveal their possible weakness, is an essential issue which has most often been omitted when designing chaotic cryptosystems.

As a result, it is well admitted that, if potential applications of dynamical systems is sought for cryptography, deeper insights are really necessary. This is the main objective of this work. More precisely, the aim of this PhD thesis is threefold.

– bringing out a connection between chaotic and conventional cryptography by comparing the respective algorithms proposed so far, highlighting the most relevant chaos-based algorithms and finding out the ones which still make sense bearing in mind that chaos turns into discrete chaos if digital implementation is sought, the dynamics being thereby closely related to pseudo-randomness. The investigation will focus on structural consideration and control theoretical concepts will be the central tools to this end.

– motivating the use of hybrid dynamical systems for the design of cryptographic

primitives. Indeed, an interesting, even though very general, cryptosystems design principle suggests mixing algebraic domains and using primitives built from combinations of boolean and arithmetic operations [LM91][KS04]. It turns out that hybrid systems is a typical class of dynamical systems which fulfils such a constraint since they involve several algebraic models which are switched in time according to some logical rules. A special class of hybrid systems will be considered : the switched linear systems.

– deriving cryptanalytic methodologies for assessing the security of cryptosystems based on hybrid systems. Again, concepts borrowed from control theory, namely identifiability and identification, will be considered. A major specificity related to the special context of secure communications and cryptography must be taken into account. In usual control theory, the variables are assumed to take values in a continuum (often $\mathbb{R}^n$ or a subset of $\mathbb{R}^n$) since they are related to physical quantities. In the cryptographic context, variables take values in finite cardinality sets (e.g. finite fields).

The layout of this manuscript is the following :

**Chapter 1** recalls the essential concepts borrowed in control theory. The thesis focuses on the switched linear discrete-time systems (switched systems for short) and three central properties, namely invertibility, flatness and stability. These properties will appear as useful to design cryptographic primitives.

**Chapter 2** introduces the general principles for scrambling information with chaotic systems. The recovery of information at the receiver part is known as chaos synchronization. Note that only the discrete-time chaotic maps are surveyed. The most important schemes obeying such a principle are surveyed, such as : additive masking, chaotic switching, parameter modulation, two channel transmission and message-embedding. The message-embedding scheme is very attractive insofar as the synchronization between the transmitter and the receiver can be guaranteed without any restriction on the rate of variation of the information to be encrypted. The chapter ends up by giving a numerical example which illustrates how to incorporate the control theory concepts described in Chapter 1 into the message-embedding cryptographic scheme.

**Chapter 3** considers the message-embedding scheme when implemented on finite state machines. All the variables of this scheme will take values in a finite field. Considering finite fields rather than the field of real numbers will deserve special treatments especially the identification technique for assessing the security of the message-embedding scheme.

**Chapter 4** introduces background on standard symmetric cryptography especially stream ciphers. Then, a comparison between the message-embedding and general class of standard symmetric cryptosystems is carried out. The major re-

sult states that, under flatness condition, the message-embedding scheme acts as a self-synchronizing stream cipher. Furthermore, it is shown how the identifiability and identification concepts are related to the security of the resulting cipher. The security is assessed in terms of the algebraic complexity of the identification process.

The **Conclusion** sums up the main contribution of this thesis and addresses open issues and possible perspectives.

**Appendix** consists of 4 parts. Appendix A recalls the numerical method for computing Lyapunov Exponents. Appendix B gives background on algebra involved in Chapter 3. Appendix C details the Gaussian elimination algorithm. Appendix D recalls the specifications of the self-synchronizing stream cipher Moustique's family.

The papers published related to this research work are listed below :

1. P. VO-TAN, G. Millérioux, J. Daafouz, Left invertibility, flatness and identifiability of switched linear dynamical systems : a framework for cryptographic applications, *International Journal of Control*, 83(1) :145-153, 2010.

2. P. Vo-Tan, G. Millérioux, J. Daafouz, A comparison between the message embedded cryptosystem and the self-synchronous stream cipher Mosquito, *18th European Conference on Circuit Theory and Design, ECCTD'2007*, 2007.

3. P. Vo-Tan, G. Millérioux, J. Daafouz, Invertibility, flatness and identifiability of switched linear dynamical systems : an application to secure communications, *47th IEEE Conference on Decision and Control, CDC'08*, 2008.

4. P. Vo-Tan, G. Millérioux, J. Daafouz, Sur les propriétés structurelles des systèmes dynamiques pour le chiffrement, *3èmes Journées Doctorales / Journées Nationales MACS*, 2009.

# Chapter 1

# Control theoretical concepts

The aim of this chapter is to provide the essential concepts in control theory which are necessary for the understanding of the remaining part of this thesis. Invertibility, flatness and stability of switched linear discrete-time system are the main topics discussed here.

By switched linear systems we mean a set of linear subsystems and a switching function which determines at each instant of time the active subsystem. This important class of hybrid systems have received a great amount of attention in the last years. From the very beginning until now, several new techniques for stability analysis and control design has appeared, as for instance those concerning switching control [Lib03], stability [LM99] [DRI02][DM02], observability [BE04][BBBV02] and identification [JHFT$^+$05]. In addition, left invertibility, which stands for the ability to recover the input sequences from the output sequences, attracted great attention. The original works [BM65][Sil69][SM69] are dedicated to linear time invariant systems. Recently, several contributions have been proposed for switched linear systems as in [SH06][MD07] for the discrete-time setting and [VL08][TL08] for the continuous-time one. Under the assumption that the switching function is known in real time, the results in [SH06][MD07] allow to recover the input sequences using an inverse system. Moreover, it is shown in [MD07] that flatness property plays a key role in deriving an explicit input/output relationship. An exhaustive presentation of flatness is out of the scope of this work but one can refer to [FLMR95][SRA04] for more details. As it turns out, the contributions related to invertibility and flatness of switched linear systems are useful in the construction of cryptographic primitives.

The outline of this chapter is as follows : Section 1.1 is dedicated to general definitions. We expose in Section 1.2 invertibility and flatness in the context of switched linear system. The corresponding detailed results can be found in [MD07]. Numerical illustrations are given in Section 1.3 before a conclusion.

## 1.1 General definitions

Let us consider the discret-time dynamical system described by the general form :

$$\begin{cases} x_{k+1} &= f_\theta(x_k, m_k) \\ y_k &= h_\theta(x_k, m_k) \end{cases} \tag{1.1}$$

Such a dynamical system is described by the 5-tuple $(A, B, X, f_\theta, h_\theta)$ where

- $k$ is the discret-time
- $A$ is the set of inputs $m_k$
- $B$ is the set of outputs $y_k$
- $X$ is the set of internal states $x_k$ also called state vectors
- $f_\theta : X \times A \longrightarrow X$ is the (next) state transition function
- $h_\theta : X \times A \longrightarrow B$ is the output function.
- $\theta$ is the parameter vector associated to the next state transition and the output function.

Throughout this chapter, the sets $A, B, X$ will be respectively $\mathbb{R}^m$, $\mathbb{R}^p$ and $\mathbb{R}^n$. $m$, $p$ and $n$ will correspond respectively to the number of inputs, outputs and the dimension of the system.

### 1.1.1 Iterated functions

**Definition 1** *The $i$-order iterated next-state function, $f_\theta^{(i)} : X \times A^i \longrightarrow X$ describes the way how the internal state $x_{k+i} \in X$ of (1.1) at time $k+i$ depends on the state $x_k \in X$ and on the sequence of $i$ input symbols $m_k \cdots m_{k+i-1} \in A^i$. It is defined for $i \geq 1$ and recursively obeys for $k \geq 0$,*

$$\begin{cases} f_\theta^{(1)}(x_k, m_k) = f_\theta(x_k, m_k) \\ f_\theta^{(i+1)}(x_k, m_k \cdots m_{k+i}) = f_\theta\Big(f_\theta^{(i)}(x_k, m_k \cdots m_{k+i-1}), m_{k+i}\Big) \quad \text{for } i \geq 1 \end{cases}$$

**Definition 2** *The $i$-order iterated output function $h_\theta^{(i)} : X \times A^{i+1} \longrightarrow B$ describes the way how the output $y_{k+i}$ of (1.1) at time $t+i$ depends on the state $x_k \in X$ and on the sequence of $i+1$ input symbols $m_k \cdots m_{k+i} \in A^{i+1}$. It is defined for $i \geq 0$ and recursively obeys for $k \geq 0$,*

$$\begin{cases} h_\theta^{(0)}(x_k, m_k) = h_\theta(x_k, m_k), \\ h_\theta^{(i)}(x_k, m_k \ldots m_{k+i}) = h_\theta\Big(f_\theta^{(i)}(x_k, m_k \cdots m_{k+i-1}), m_{k+i}\Big) \quad \text{for } i \geq 1 \end{cases}$$

### 1.1.2 Relative degree

For a Single Input Single Output (SISO) system (that is $m = p = 1$), we define the relative degree as follows :

**Definition 3** *The relative degree of the dynamical system (1.1) is the quantity denoted $R$ with*

- $R = 0$ *if* $\exists x_k \in X,\ \exists m_k, m'_k \in A$ *with* $h_\theta(x_k, m_k) \neq h_\theta(x_k, m'_k)$.
  *In other words, there exists a state $x_k \in X$ and two distinct input symbols $m_k, m'_k \in A$ that lead to different values of the output.*

- $R > 0$ *if for any sequence $m_{k+1} \cdots m_{k+R}$ of input symbols*

$$\exists x_k \in X,\ \exists m_k, m'_k \in A \text{ with}$$

$$h_\theta^{(R)}(x_k, m_k \cdots m_{k+R}) \neq h_\theta^{(R)}(x_k, m'_k \cdots m'_{k+R})$$

  *In others words, for $i < R$, the iterated output function $h_\theta^{(i)}$ only depends on $x_k$ while for $i \geq R$, it depends both on $x_k$ and on the sequence of $i - R + 1$ input symbols $m_k \cdots m_{k+i-R}$. In particular, for $i = R$, the iterated output function depends both on $m_k$ and on $x_k$, that is, there exists a state $x_k \in X$ and two distinct input symbols $m_k \in A$ and $m'_k \in A$ that lead to different values of the output, for any sequence $m_{k+1} \cdots m_{k+R}$ of input symbols.*

Roughly speaking the relative degree of the dynamical system (1.1) is the minimum number of iterations such that the output at time $k + R$ is influenced by the input at time $k$. Consequently, for $R > 0$, the $R$-order output function $h_\theta^{(R)}$ may be considered as a function on $X \times A \to B$.

Finally, one has for $R \geq 0$ :

$$y_{k+R} = h_\theta^{(R)}(x_k, m_k) \tag{1.2}$$

### 1.1.3 Left invertibility

**Definition 4** *The dynamical system (1.1) is* left invertible *if there exists a non-negative integer $r < \infty$, called* inherent delay*, such that for any two inputs $m_k \in A$ and $m'_k \in A$ the following implication holds :*

$$\begin{array}{l} \forall x_k \in X \\ h^{(0)}(x_k, m_k) \cdots h^{(r)}(x_k, m_k \cdots m_{k+r}) = h^{(0)}(x_k, m'_k) \cdots h^{(r)}(x_k, m'_k \cdots m'_{k+r}) \\ \Rightarrow m_k = m'_k. \end{array} \tag{1.3}$$

The left invertibility property means that the input $m_k$ is uniquely determined by the knowledge of the state $x_k$ and of the output sequence $y_k, \ldots, y_{k+r}$.

For dynamical systems with SISO and when $A = B$, another interpretation of left invertibility is that for any internal state $x_k \in X$, the map

$$h_{x_k} : \begin{array}{ccc} A & \longrightarrow & A \\ m_k & \longmapsto & h^{(r)}(x_k, m_k) \end{array} \tag{1.4}$$

is a bijection.

### 1.1.4 Flatness

Flatness was introduced by Fliess *and al.* [FLMR95] in 1995 and a deep insight into the subject can be found in the quite recent book [SRA04].

**Definition 5** *An output for (1.1) is said to be* flat *if all system variables of (1.1) can be expressed as a function of $y_k$ and a finite number of its forward/backward iterates. In particular, there exists two functions $F$ and $G$ and integers $t_1 < t_2$ and $t'_1 < t'_2$ such that*

$$\begin{array}{rcl} x_k & = & F(y_{k+t_1}, \cdots, y_{k+t_2}) \\ m_k & = & G(y_{k+t'_1}, \cdots, y_{k+t'_2}) \end{array} \tag{1.5}$$

Then, the dynamical system (1.1) is said to be *flat* if it admits a flat output and the *flatness characteristic number* is defined as the quantity $t_2 - t_1 + 1$.

## 1.2 Particularization for switched linear systems

We examine switching linear discrete-time systems of the form :

$$\left\{ \begin{array}{rcl} x_{k+1} & = & A_{\sigma(k)} x_k + B_{\sigma(k)} m_k \\ y_k & = & C_{\sigma(k)} x_k + D_{\sigma(k)} m_k \end{array} \right. \tag{1.6}$$

where $x_k \in \mathbb{R}^n$, $m_k \in \mathbb{R}^m$ and $y_k \in \mathbb{R}^p$ are the states, the inputs and the measurements, respectively. All the matrices, namely $A_{\sigma(k)} \in \mathbb{R}^{n\times n}$, $B_{\sigma(k)} \in \mathbb{R}^{n\times m}$, $C_{\sigma(k)} \in \mathbb{R}^{p\times n}$ and $D_{\sigma(k)} \in \mathbb{R}^{p\times m}$ belong to the respective finite sets $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ and $(D_j)_{1\leq j\leq J}$. At a given time $k$, the index $j$ corresponds to the mode of the system and results from a switching function $\sigma : k \in \mathbb{N} \mapsto j = \sigma(k) \in \{1, \ldots, J\}$. $\{\sigma\}_{k_1}^{k_1+T}$ refers to the mode sequence $\{\sigma(k_1), \ldots, \sigma(k_1 + T)\}$. For a given switching rule $\sigma$, the set of corresponding mode sequences over any interval of time of length $T + 1$ is denoted by $\Sigma^T$. This set may contain either all possible mode sequences (also called paths) if there are not any repetitive switching patterns or can be reduced if any. We assume that the mode is known, either accessible or reconstructed (see [BBBV02] for this reconstruction issue). No restriction on the time separation between switches ("dwell time") is imposed.

At time $k$, for each initial state $x_k \in \mathbb{R}^n$, when the system (1.6) is driven by the input sequence $\{m\}_k^{k+T} = \{m_k, \ldots, m_{k+T}\}$, for a mode sequence $\{\sigma\}_k^{k+T}$, $\{x(x_k, \sigma, m)\}_k^{k+T}$ refers to the solution in the interval of time $[k, k+T]$ of (1.6) emanating from $x_k$ and $\{y(x_k, \sigma, m)\}_k^{k+T}$ refers to the corresponding output sequence in the same interval of time $[k, k+T]$.

We recall in this Subsection some important results borrowed from the paper [MD09].

We first recall some notations related to specific vectors and matrices.

For $i < 0$ :
$M^i_{\sigma(k)} = \mathbf{0}$

For $i = 0$ :
$M^0_{\sigma(k)} = D_{\sigma(k)}$

For $i > 0$ :

$$M^i_{\sigma(k)} = \begin{pmatrix} D_{\sigma(k)} & \mathbf{0}_{p\times m} & \cdots & \cdots & \cdots \\ C_{\sigma(k+1)}B_{\sigma(k)} & D_{\sigma(k+1)} & \mathbf{0}_{p\times m} & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \\ C_{\sigma(k+i)}A^{\sigma(k+i-1)}_{\sigma(k+1)}B_{\sigma(k)} & C_{\sigma(k+i)}A^{\sigma(k+i-1)}_{\sigma(k+2)}B_{\sigma(k+1)} & \cdots & C_{\sigma(k+i)}B_{\sigma(k+i-1)} & D_{\sigma(k+i)} \end{pmatrix} \tag{1.7}$$

with the direct transition matrix

$$A^{\sigma(k_1)}_{\sigma(k_0)} = \begin{cases} A_{\sigma(k_1)}A_{\sigma(k_1-1)}\ldots A_{\sigma(k_0)} & \text{if } k_1 \geq k_0 \\ \mathbf{1}_n & \text{if } k_1 < k_0 \end{cases}$$

$$\bar{I}_m = (\mathbf{1}_m \ \mathbf{0}) \tag{1.8}$$

$$\mathcal{O}^i_{\sigma(k)} = \begin{pmatrix} C_{\sigma(k)} \\ C_{\sigma(k+1)}A_{\sigma(k)} \\ \vdots \\ C_{\sigma(k+i)}A^{\sigma(k+i-1)}_{\sigma(k)} \end{pmatrix} \tag{1.9}$$

$$\underline{m}^i_k = \begin{pmatrix} m_k \\ m_{k+1} \\ \vdots \\ m_{k+i} \end{pmatrix}, \quad \underline{y}^i_k = \begin{pmatrix} y_k \\ y_{k+1} \\ \vdots \\ y_{k+i} \end{pmatrix} \tag{1.10}$$

When (1.6) is driven by an input sequence $\{m\}_k^\infty$ and a mode sequence $\{\sigma\}_k^\infty$, one has for all $i \geq 0$ :

$$\underline{y}^i_k = \mathcal{O}^i_{\sigma(k)}x_k + M^i_{\sigma(k)}\underline{m}^i_k \tag{1.11}$$

### 1.2.1 Input left invertibility of a switched linear system

For switched linear systems, the issue of recovering the input of a dynamical system from the output, assuming that the switching sequence is known is called the input left inversion. In spite of the fact that left invertibility refers to the ability of both recovering the input and the mode, input left invertibility is also sometimes but abusively merely called left invertibility.

**Theorem 1** *[MD09] The system (1.6) is input left invertible if there exists a nonnegative integer $r < \infty$ such that for all mode sequences in $\Sigma^r$,*

$$rank\ M^r_{\sigma(k)} - rank\ M^{r-1}_{\sigma(k+1)} = m \tag{1.12}$$

*The quantity $r$ is called left inherent delay*

### 1.2.2 Input left inversion of a switched linear system

**Definition 6** *A system is a left $r$-delay inverse for (1.6) if, under identical initial conditions $x_0$ and identical mode sequences $\{\sigma\}_0^\infty$, when driven by $\underline{y}^r_k$, its output $\hat{m}_{k+r}$ fulfills $\hat{m}_{k+r} = m_k$ for all $k \geq 0$*

**Theorem 2** *[MD09] Assume that (1.6) is input left invertible with left inherent delay $r$. The system*

$$\left\{ \begin{array}{rcl} \hat{x}_{k+r+1} & = & P^r_{\sigma(k)}\hat{x}_{k+r} + B_{\sigma(k)}\bar{I}_m M^{r\dagger}_{\sigma(k)}\underline{y}^r_k \\ \hat{m}_{k+r} & = & -\bar{I}_m M^{r\dagger}_{\sigma(k)}\mathcal{O}^R_{\sigma(k)}\hat{x}_{k+r} + \bar{I}_m M^{r\dagger}_{\sigma(k)}\underline{y}^r_k \end{array} \right. \tag{1.13}$$

*with*

$$P^r_{\sigma(k)} = A_{\sigma(k)} - B_{\sigma(k)}\bar{I}_m M^{r\dagger}_{\sigma(k)}\mathcal{O}^r_{\sigma(k)} \tag{1.14}$$

*is a left $r$-delayed inverse system for (1.6) with $\bar{I}_m = (\mathbf{1}_m\ \mathbf{0}_{m\times(m\cdot r)})$.*

**Remark 1** *In the Definition 6, the initial condition is considered at the particular discrete time $k = 0$ but can be replaced by any other initial condition $x_k$ considered at the discrete time $k = k_0$ and $\hat{m}_{k+r} = m_k$ for all $k \geq\ k_0$.*

**Remark 2** *It is also shown in [MD09] that the state vector of the left $r$-delay inverse (1.13) fulfills $\hat{x}_{k+r} = x_k$ for all $k \geq 0$ and that the error of reconstruction $\epsilon_k = x_k - \hat{x}_{k+r}$ fulfills*

$$\epsilon_{k+1} = P^r_{\sigma(k)}\epsilon_k \tag{1.15}$$

### 1.2.3 Flatness of a switched linear system

#### 1.2.3.1 Algebraic characterization

Let us define the inverse transition matrix as

$$P^{\sigma(k_1)}_{\sigma(k_0)} = \begin{cases} P^r_{\sigma(k_1)} P^r_{\sigma(k_1-1)} \cdots P^r_{\sigma(k_0)} & \text{if } k_1 \geq k_0 \\ \mathbf{1}_n & \text{if } k_1 < k_0 \end{cases}$$

with

$$P^r_{\sigma(k)} = A_{\sigma(k)} - B_{\sigma(k)} \bar{I}_m M^{r\dagger}_{\sigma(k)} \mathcal{O}^r_{\sigma(k)} \tag{1.16}$$

**Theorem 3** *[MD09] A componentwise independent output $y_k$ of the system (1.6) assumed to be square ($m = p$) and left input invertible with inherent delay $r$, is a* flat output *if there exists a positive integer $K < \infty$ such that, for all mode sequences in $\Sigma^{r+K-1}$, the following equality applies for all $k \geq 0$ :*

$$P^{\sigma(k+K-1)}_{\sigma(k)} = \mathbf{0} \tag{1.17}$$

$\Sigma^{r+K-1}$ stands for the set of mode sequences over the interval of time $[k, \ldots, k + r + K - 1]$.

#### 1.2.3.2 Other characterization

In order to test whether a system is flat, we can also try to express each component $x^{(i)}_k$ $(i = 1, \ldots, n)$ and $m^{(i)}_k$ $(i = 1, \ldots, m)$ as a function which depends only on the output and its iterations. The method can be based on the elimination technique. There exists many elimination algorithms (cf. [Wan91] for a detailed comparison) especially derived from resultants theory, characteristic set or Gröbner basis. We are generally not able to a priori decide which method is the best. The elimination algorithms are incorporated in many symbolical computation software as Maple, Mathematica or the freeware Maxima [1] which is based on the theory of resultant [Wan91].

### 1.2.4 Stability of switched linear systems

We recall two important theorems characterizing the stability of switched linear systems. Consider (1.6) in the autonomous regime :

$$x_{k+1} = A_{\sigma(k)} x_k \tag{1.18}$$

1. available at http ://maxima.sourceforge.net

**Theorem 4 (Quadratic stability)** *[BGFB94] The system* (1.18) *is quadratically stable if and only if there exists a symmetric positive definite matrix $P$ of dimension $n$, such that :*

$$\begin{pmatrix} P & A_i^T P \\ PA_i & P \end{pmatrix} > 0 \quad \forall i \in \{1, \ldots, J\} \tag{1.19}$$

The relation (1.19) consists of $J$ matrix inequalities called LMIs (Linear Matrix Inequalities) where $P$ is the indeterminated matrix. If Theorem 4 is fulfilled, we can show that the Lyapunov function

$$V(x_k) = x_k^T P x_k$$

fulfills :

$$\Delta V(x_{k+1}, x_k) = V(x_{k+1}) - V(x_k) = x_{k+1}^T P x_{k+1} - x_k^T P x_k < 0, \quad \forall x_k \neq 0$$

**Theorem 5 (Poly-quadratic stability)** *[DRI02] The system* (1.18) *is poly-quadratically stable if and only if there exists symmetric positive definite matrices $S_i$ and matrices $G_i$ of appropriate dimensions, such that :*

$$\begin{pmatrix} G_i + G_i^T - S_i & G_i^T A_i^T \\ A_i G_i & S_j \end{pmatrix} > 0 \tag{1.20}$$

*for all $i \in \{1, \ldots, J\}$ and $j \in \{1, \ldots, J\}$.*

In this case, the Lyapunov function :

$$V(x_k) = x_k^T P_{\sigma(k)} x_k$$

where

$$\sigma(k) \in \{1, \ldots, J\} \quad \text{and} \quad P_i = S_i^{-1} \quad i \in \{1, \ldots, J\}$$

verifies :

$$\begin{aligned} \Delta V(x_{k+1}, x_k) &= V(x_{k+1}) - V(x_k) \\ &= x_{k+1}^T P_{\sigma(k+1)} x_{k+1} - x_k^T P_{\sigma(k)} x_k < 0, \quad \forall x_k \neq 0 \end{aligned}$$

## 1.3 Example

In this section, we illustrate the theoretical concepts of left invertibility and flatness presented in this chapter. We consider the system (1.6) with the special setting :

$$\begin{aligned} A_{\sigma(k)} &= \begin{pmatrix} q_{\sigma(k)}^{(1)} & 1 \\ 0.5 & 0 \end{pmatrix} & B_{\sigma(k)} &= \begin{pmatrix} 0 \\ q_{\sigma(k)}^{(2)} \end{pmatrix} \\ C_{\sigma(k)} &= (1 \ 0) & D_{\sigma(k)} &= 0 \end{aligned} \tag{1.21}$$

and the number of modes $J = 4$. The time-varying entries fulfill $q_1^{(1)} = q_2^{(1)} = 1.7$, $q_3^{(1)} = q_4^{(1)} = -1.7$, $q_1^{(2)} = q_3^{(2)} = -0.01$, $q_2^{(2)} = q_4^{(2)} = 0.01$.

**Inherent delay**

For evaluating the inherent delay, we construct the matrices $M^i_{\sigma(k)}$ as in (1.7). We get that :

$$\begin{aligned}
M^0_{\sigma(k)} &= D_{\sigma(k)} = 0 \\
M^1_{\sigma(k)} &= \begin{pmatrix} D_{\sigma(k)} & 0 \\ C_{\sigma(k+1)}B_{\sigma(k)} & D_{\sigma(k+1)} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \\
M^2_{\sigma(k)} &= \begin{pmatrix} D_{\sigma(k)} & 0 & 0 \\ C_{\sigma(k+1)}B_{\sigma(k)} & D_{\sigma(k+1)} & 0 \\ C_{\sigma(k+2)}A_{\sigma(k+1)}B_{\sigma(k)} & C_{\sigma(k+1)}B_{\sigma(k)} & D_{\sigma(k+2)} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ q^{(2)}_{\sigma(k)} & 0 & 0 \end{pmatrix}
\end{aligned}$$

It turns out that

$$rank\ M^2_{\sigma(k)} - rank\ M^1_{\sigma(k+1)} = 1$$

and so, the inherent delay is $r = 2$ according to Theorem 1.

**Flatness characterization based on the algebraic approach**

$$A_{\sigma(k)} = \begin{pmatrix} q^{(1)}_{\sigma(k)} & 1 \\ 0.5 & 0 \end{pmatrix} \quad A_{\sigma(k+1)} = \begin{pmatrix} q^{(1)}_{\sigma(k+1)} & 1 \\ 0.5 & 0 \end{pmatrix}$$

Hence the direct transition matrix reads :

$$A^{\sigma(k+1)}_{\sigma(k)} = \begin{pmatrix} q^{(1)}_{\sigma(k+1)} & 1 \\ 0.5 & 0 \end{pmatrix} \begin{pmatrix} q^{(1)}_{\sigma(k)} & 1 \\ 0.5 & 0 \end{pmatrix} = \begin{pmatrix} q^{(1)}_{\sigma(k+1)}q^{(1)}_{\sigma(k)} + 0.5 & q^{(1)}_{\sigma(k+1)} \\ 0.5q^{(1)}_{\sigma(k)} & 0.5 \end{pmatrix}$$

and one gets :

$$\mathcal{O}^2_{\sigma(k)} = \begin{pmatrix} C_{\sigma(k)} \\ C_{\sigma(k+1)}A_{\sigma(k)} \\ C_{\sigma(k+2)}A^{\sigma(k+1)}_{\sigma(k)} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ q^{(1)}_{\sigma(k)} & 1 \\ q^{(1)}_{\sigma(k+1)}q^{(1)}_{\sigma(k)} + 0.5 & q^{(1)}_{\sigma(k+1)} \end{pmatrix}$$

We are now computing the pseudo inverse of matrix

$$M^2_{\sigma(k)} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ q^{(2)}_{\sigma(k)} & 0 & 0 \end{pmatrix}$$

The singular value decomposition of $M^2_{\sigma(k)}$ yields :

$$M^2_{\sigma(k)} = USV^T = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} q^{(2)}_{\sigma(k)} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Consequently, the pseudo inverse of $M^2_{\sigma(k)}$ reads :

$$M^{2\dagger}_{\sigma(k)} = VS^{\dagger}U^T = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \frac{1}{q^{(2)}_{\sigma(k)}} \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix} = (q^{(2)}_{\sigma(k)})^{-1} \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix}$$

The computation of (1.14) gives :

$$\begin{aligned} P^2_{\sigma(k)} &= A_{\sigma(k)} - B_{\sigma(k)} \bar{I}_m M^{2\dagger}_{\sigma(k)} \mathcal{O}^2_{\sigma(k)} \\ &= \begin{pmatrix} q^{(1)}_{\sigma(k)} & 1 \\ -q^{(1)}_{\sigma(k+1)} q^{(1)}_{\sigma(k)} & -q^{(1)}_{\sigma(k+1)} \end{pmatrix} \end{aligned}$$

and

$$P^2_{\sigma(k+1)} = \begin{pmatrix} q^{(1)}_{\sigma(k+1)} & 1 \\ -q^{(1)}_{\sigma(k+2)} q^{(1)}_{\sigma(k+1)} & -q^{(1)}_{\sigma(k+2)} \end{pmatrix}$$

This yields

$$P^{\sigma(k+1)}_{\sigma(k)} = P^2_{\sigma(k+1)} P^2_{\sigma(k)} = \mathbf{0}$$

As a result, (1.21) is flat with $K = 2$ according to Theorem 3.

**Flatness characterization based on elimination theory**
We use the freeware Maxima [2] in this example.

There are three necessary steps :
– define the state equations
– write out the successive iterations of the state equation
– choose the variables to be eliminated

For the first component $x^{(1)}_k$, the answer is obvious and there is no need for supplementary computations. We have :

$$x^{(1)}_k = y_k \tag{1.22}$$

To understand the script Maxima for searching the expression of the second component $x^{(2)}_k$ and of $m_k$, let us introduce some notations. The variables $x^{(i)}_k$, $y_k$ and $m_k$ are respectively written as `xik`, `yk` and `mk`. The $l$-th iteration is denoted `xikl`, `ykl` and `mkl`. An equation always has to be labelled like `eqj` where `j` stands for the $j$-th iteration. The non-constant entries of matrices $A_{\sigma(k)}$ and $B_{\sigma(k)}$, that are $q^{(1)}_{\sigma(k)}$ and $q^{(2)}_{\sigma(k)}$ are denoted `q1k` and `q2k`. Their $l$-th iterations are denoted `q1kl` and `q2kl`. Finally, an equation is implicitly known as a relation which is equal to zero. Now we detail the different scripts.

We precise firstly the state equations of the system :

2. available at http ://maxima.sourceforge.net

```
(%i1) eq1:x1k1-q1k*x1k-x2k;
(%i2) eq2:x2k1-0.5*x1k-q2k*mk;
(%i3) eq3:yk-x1k;
```

We write out the iterations of the state equations :

```
(%i4) eq4:x1k2-q1k1*x1k1-x2k1
(%i5) eq5:x2k2-0.5*x1k1-q2k1*mk1;
(%i6) eq6:yk1-x1k1;
(%i7) eq7:yk2-x1k2;
```

To obtain the expression (if it exists) $x_k^{(2)}$ as a unique function of the output and its possible iterations, we use the function `eliminate` in Maxima which is based on the resultants technique for which the list of used equations and eliminated variables must be provided :

```
eliminate([eq1,eq2,eq3,eq4,eq5,eq6,eq7],
[mk,mk1,x1k1,x1k2,x2k1,x2k2]);
```

We get that :

$$x_k^{(2)} = y_{k+1} - q_{\sigma(k)}^{(1)} y_k \tag{1.23}$$

To obtain the expression (if it exists) of $m_k$, we give the list of used equations and the eliminated variables :

```
eliminate([eq1,eq2,eq3,eq4,eq5,eq6,eq7],
[mk1,x2k,x1k1,x1k2,x2k1,x2k2]);
```

We obtain :

$$2q_{\sigma(k)}^{(2)} m_k = 2y_{k+2} - 2q_{\sigma(k+1)}^{(1)} y_{k+1} - y_k \tag{1.24}$$

From (1.22), (1.23) and (1.24), we conclude that the two components $x_k^{(i)}$ $(i = 1, 2)$ of the state vector and the input $m_k$ depend exclusively on the output $y_k$ and its iterates. We so conclude that the system (1.21) is flat.

## 1.4 Conclusion

In this chapter, key results for switched linear systems further required for our purpose, namely the context of secure communications and cryptography, have been recalled. The main ones concern structural properties, namely left invertibility and flatness.

# Chapter 2

# Chaos-based secure communication

This chapter is devoted to a review of the most popular chaos-based cryptosystems proposed over the years since the 90's. Two main approaches can be distinguished. The first one consists in computing a great number of iterations of a chaotic map, using a digital message as initial data. We refer to [SAMK05] [Sch01] [HNSM91] [ASK05] and references therein for details. The second method, which is discussed in this chapter, is based on signals synchronization (see the reviews [Ogo93] [Has98] [Yan04] [AL06] [MAD08] according to the chronology). Chaos-based schemes with their advantages and drawbacks are presented here : additive masking, chaotic switching, parameter modulation, two-channel transmission and message-embedding.

The outline of this chapter is as follows : We start, in Section 2.1, by giving a formal definition of chaos. Section 2.3 is dedicated to chaos-based cryptosystems listed above. We give in Section 2.4 a numerical experiment of the message-embedding scheme, which highlights the role played by control theoretic concepts described in Chapter 1. We end this chapter with a conclusion.

## 2.1 Background on chaos

Autonomous nonlinear dynamics corresponding to maps can be written in the generic explicit form :

$$x_{k+1} = f(x_k), \qquad \text{with the initial condition } x_0 \tag{2.1}$$

$x_k \in \mathbb{R}^n$ is called the state vector and $n$ corresponds to the dimension of the system.

The system is said to be *autonomous* since the discrete time $k$ does not appear explicitly in the equation (2.1).

The solution of (2.1) from the initial condition $x_0$, is a sequence of points called *iterated sequence*, or *discrete phase trajectory*, or *orbit*. The time evolution of the state is completely determined by the initial state $x_0$ of the system and the dynamics. In general, the explicit solution of (2.1) in terms of known elementary and transcendental functions is unknown but actually, we are often only interested in the steady-state behaviors. The iterated sequence may reach more or less complex steady states which can coexist, the most often encountered being :

**Stationary state**
A stationary state is also called equilibrium point or fixed points. It obeys :
$x_{k+1} = x_k = x^*$

**Periodic orbit**
A periodic orbit corresponds to cycles of finite order $K^o$ and obeys :
$x_{k+K^o} = x_k$ and $x_{k+K'^o} \neq x_k$ for $K'^o < K^o$

**Chaotic orbit**
A chaotic orbit can be viewed in some sense as an infinite period trajectory and thus obeys :

$x_{k+K^o} \neq x_{K^o} \ \ \forall K^o$ and $x_k$ is bounded.

This relation is not sufficient to formally define chaos. In the following, we give a strict definition of chaos.

Let $(I \in \mathbb{R}, l)$ denote a compact metric space ($l$ is a distance) and consider the nonlinear continuous function defining the map :

$$f : I \to I, \quad x_{k+1} = f(x_k), \quad x_0 \in I$$

Before providing a strict definition of chaos which is due to R.L. Devaney [Dev89], some basic definitions are required.

**Definition 7** *$f$ is said to have the property of* sensitive dependence on initial conditions *or to be* sensitive *to initial conditions if there exists some $\delta > 0$ such that, for any $x_0 \in I$ and any $\epsilon > 0$, there is a point $y_0 \in I$ and an integer $j \geq 0$ fulfilling*

$$l(x_0, y_0) > \epsilon \Rightarrow d(f^{(j)}(x_0), f^{(j)}(y_0)) > \delta$$

where $l$ stands for the distance and $f^{(j)}$ for the $j$-order iterated of $f$.

**Definition 8** *$f$ is said to be* topologically transitive *if, $U$ and $V$ being non-empty open sets in $I$, there is some $x_0 \in U$ and an index $j \in \mathbb{Z}^+$ such that $f^{(j)}(x_0) \in V$ or equivalently, there exist an index $j \in \mathbb{Z}^+$ such that $f^{(j)}(U) \cap V \neq \emptyset$*

We are now in position of stating the definition of a chaotic system in the sense of Devaney.

**Definition 9** *A continuous function $f : I \to I$ is said to be a* chaotic map *or to define a* chaotic dynamical system *if :*
*a) $f$ is sensitive to initial conditions*
*b) $f$ is topologically transitive*
*c) the set of periodic points of $f$ is dense in $I$*

To determine the sensitivity to initial conditions of a dynamical system, we can resort to the notion of Lyapunov Exponents. It is based on the measure of the divergence rate between two distinct trajectories which started from two nearby initial conditions (see Appendix A for a numerical routine for the computation of Lyapunov Exponents)

## 2.2 Some examples of discrete chaotic maps

**Logistic chaotic map**
The Logistic chaotic map has been considered by the biologist Robert May [May76] for describing the evolution of the population of a species. It is a one dimensional map which is given as :

$$x_{k+1} = \theta x_k(1 - x_k)$$

with $0 < \theta \leq 4$. The state vector $x_k \in [0, 1]$ is the number of individuals in the population at instant $k$ and the parameter $\theta$ stands for the increment factor of the population.

**Henon chaotic map**
The Henon chaotic map [Hen76] is a two-dimensional map given by :

$$\begin{cases} x_{k+1}^{(1)} = -1.4(x_k^{(1)})^2 + x_k^{(2)} + 1 \\ x_{k+1}^{(2)} = 0.3x_k^{(1)} \end{cases}$$

The corresponding Henon attractor is depicted in FIG. 2.2A.

**Ikeda chaotic map**
The two-dimensional chaotic map Ikeda [Ike79], considered in optics by the physician Ikeda, are given in the following form :

$$\begin{cases} x_{k+1}^{(1)} = 1 + 0.9(x_k^{(1)} cos(\theta_k) - x_k^{(2)} sin(\theta_k)) \\ x_{k+1}^{(2)} = 0.9(x_k^{(1)} sin(\theta_k) + x_k^{(2)} cos(\theta_k)) \\ \theta_k = 0.4 - \frac{6}{1+(x_k^{(1)})^{(2)}+(x_k^{(2)})^{(2)}} \end{cases}$$

The corresponding attractor of the Ikeda map is depicted in FIG. 2.2B.

**Lozi chaotic map**
First considered in [HOP92], the Lozi chaotic map is a two-dimensional chaotic map governed by :

$$\begin{cases} x_{k+1}^{(1)} = -1.7|x_k^{(1)}|x_k^{(2)} + 1 \\ x_{k+1}^{(2)} = 0.5x_k^{(1)} \end{cases}$$

The corresponding attractor of the Lozi map is depicted in FIG. 2.1.

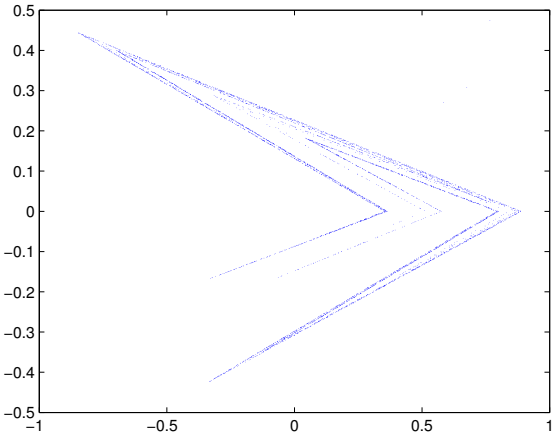


FIGURE 2.1 – The chaotic attractor of the Lozi map

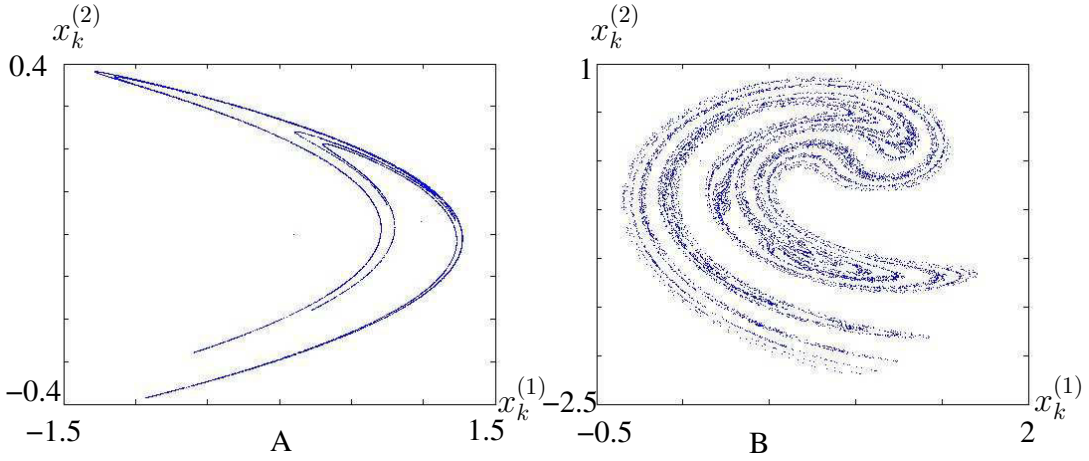


FIGURE 2.2 – Chaotic attractor. A : Henon map, B : Ikeda map.

## 2.3 General principles of scrambling information with chaotic systems

Scrambling information (or message) denoted hereafter $m_k \in \mathbb{R}$ views a chaotic system with dynamic $f$ as a complex sequence generator. $f$ is often specified by a state representation with corresponding state vector $x_k \in \mathbb{R}^n$, the dimension of the system being $n$. $f$ is parametrized by a vector $\theta$ of dimension $L$ which is expected to act as the secret key. Only a part of the state vector $x_k$ obtained via a function $h$, possibly parametrized by $\theta$ as well, called the "output" and denoted $y_k$, is conveyed through the public channel. $y_k$ is usually of low dimension and should be unidimensional in the ideal case. In what follows, we will assume that $y_k$ is a scalar (dimension 1) belonging to $\mathbb{R}$, the transmitter being thus restricted to a so-called Single Input Single Output (SISO) system. The receiver is a dynamical system with dynamics $\tilde{f}$ and with output function $\tilde{h}$, both parametrized by $\hat{\theta}$. Its state vector is denoted $\hat{x}_k$.

Both functions $\tilde{f}$ and $\tilde{h}$ must be properly chosen to recover the message $m_k$ at the receiver side. A first condition is that $\hat{\theta} = \theta$. For most of the chaos-based cryptosystems, the recovering of the message $m_k$ is performed in two steps : *chaos synchronization* and *static inversion.*

**i) Chaos synchronization**
Let $M$ be a constant matrix of appropriate dimension, and $U$ a non empty set of initial conditions. There are two main concepts of synchronization :

**Definition 10** *Asymptotical synchronization :*

$$\lim_{k\to\infty} \|Mx_k - \hat{x}_k\| = 0 \quad \forall \hat{x}_0 \in U. \tag{2.2}$$

**Definition 11** *Finite time synchronization :*

$$\exists k_f < \infty, \quad \|Mx_k - \hat{x}_k\| = 0 \quad \forall \hat{x}_0 \in U \quad \textit{and} \quad \forall k \geq k_f. \tag{2.3}$$

If only a part of the components are reconstructed, the observer is a reduced observer and $rank(M) < n$. If all the components of the state vector are reconstructed, the observer is a full observer and $M$ is the identity matrix.

**Remark 3** *In practice, if a digital implementation is carried out, because of the finite accuracy of any computers, the error of an asymptotical synchronization can be considered to be zero after a finite transient time.*

Synchronization can be viewed as a state reconstruction. In 1997 several papers [HY97] [MWO97] [Mil97] [GM97] brought out this connection. As a result, the

receiver often consists of an observer.

**ii) Static inversion**
Static inversion involves a "static" function $d$ that depends on the internal state $\hat{x}_k$ and the output $y_k$. The function $d$ delivers a quantity $d(\hat{x}_k, y_k) = \hat{m}_k$ and must verify

$$d(\hat{x}_k, y_k) := \hat{m}_k = m_k \quad \text{if} \quad \hat{x}_k = x_k$$

Various cryptosystems, corresponding to distinct ways of scrambling a message, have drawn the attention of researchers over the years. They are reviewed in the following subsections. Let us point out that we are going to restrict to discrete-time systems (maps) having in mind the comparison with digital conventional cryptography, but most of these chaotic cryptosystems can also be found in the literature for the continuous time.

### 2.3.1 Additive masking

This scheme was first suggested in [MVH93] and [WO93]. The information $m_k$ to be concealed is merely added to the output $y_k$ of the *transmitter* (FIG. 2.3) :

$$\begin{cases} x_{k+1} &= f_\theta(x_k), \\ y_k &= h_\theta(x_k) + m_k. \end{cases} \tag{2.4}$$

The generic equations of the *receiver* read :

$$\begin{cases} \hat{x}_{k+1} &= \tilde{f}_\theta(\hat{x}_k, y_k), \\ \hat{y}_k &= \tilde{h}_\theta(\hat{x}_k). \end{cases} \tag{2.5}$$

The quantity $y_k$ which appears in (2.5) reveals the unidirectional coupling between both the transmitter and the receiver systems. Provided that synchronization (2.2) or (2.3) can be achieved, the recovering of the information is performed by the static inversion

$$\hat{m}_k = y_k - \tilde{h}_\theta(\hat{x}_k).$$

Unfortunately, most often, the information cannot be exactly retrieved. Indeed, $m_k$ acts as a disturbance on the channel and precludes the *receiver* from being exactly synchronized ; neither (2.2) nor (2.3) can be exactly fulfilled. As a result, $\hat{x}_k \neq x_k$, $\hat{y}_k \neq y_k$ and, finally, $\hat{m}_k \neq m_k$ for any $k$.

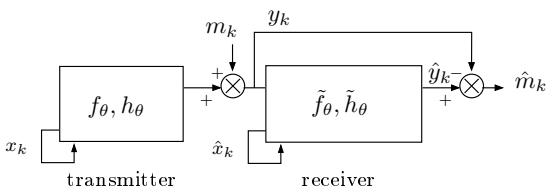


FIGURE 2.3 – Additive masking

## 2.3.2 Modulation

### 2.3.2.1 Chaotic switching

Chaotic switching is also referred to as chaotic modulation or chaos shift keying. Such a technique has been mostly proposed in the digital communications context. A thorough description can be found in [GPO98], even though the method was proposed a couple of years before, say, in 1993 [HPM93]. Basically, at the *transmitter* side, to each symbol $m_k = m^i$, belonging to a finite set $\{m^1, \ldots, m^N\}$, is assigned a chaotic signal emanating from the dynamic $f_\theta^i$ with output function $h_\theta^i$ $(i = 1, \ldots, N)$. Therefore, in the transmitter description, the index $i$ depends on $m_k$.

$$\left\{ \begin{array}{lcl} x_{k+1} & = & f_\theta^{i(m_k)}(x_k), \\ y_k & = & h_\theta^{i(m_k)}(x_k). \end{array} \right. \tag{2.6}$$

The simplest case involves binary-valued information and only two different chaotic dynamics $f_\theta^1$, $f_\theta^2$ are needed. Then, according to the current value of the symbol $m_k$ at times $k = jK_0$ $(j \in \mathbb{N})$, a switch is periodically triggered on every $K_0$ samples. During the interval of time $[jK_0, (j+1)K_0 - 1]$, $m_k$ is assumed to be constant and the chaotic signal $y_k$ of the system which has been switched on is conveyed through the channel (FIG. 2.4).

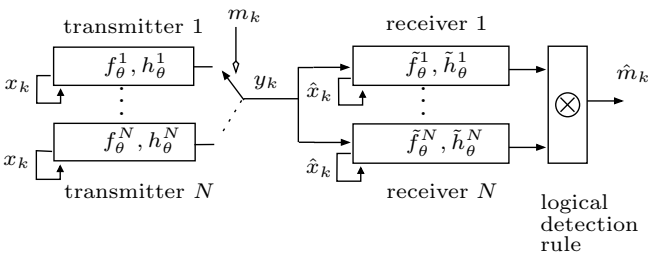


FIGURE 2.4 – Chaotic switching

The objective at the *receiver* end is to decide which chaotic system $f_\theta^i$ is most likely to have produced the sequence $\{y_k\}_{jK_0}^{(j+1)K_0-1}$. To this end, the receiver part is composed of as many systems, say $N$, as at the transmitter part :

$$\left\{ \begin{array}{lcl} \hat{x}_{k+1} & = & \tilde{f}_\theta^i(\hat{x}_k, [y_k]), \\ \hat{y}_k & = & \tilde{h}_\theta^i(\hat{x}_k). \end{array} \right. \tag{2.7}$$

The symbol $[\cdot]$ means that $y_k$ is possibly involved in $\tilde{f}_\theta^i$ and distinguishes two methods : the coherent and the non coherent detections. Non coherent detection involves statistical approaches mainly based upon correlation operations between the transmitted signal $y_k$ and the estimated signal $\hat{y}_k$. In this case, the receivers are autonomous systems with dynamics $\tilde{f}_\theta^i$, and $y_k$ must be omitted in (2.7). Coherent methods require the synchronization of both the transmitter and the

receiver. The synchronization (2.2) or (2.3) (where $\hat{x}_0$ must be replaced by $\hat{x}_{jK_0}$) is obtained by unidirectional coupling through the variable $y_k$ which is really involved in $\tilde{f}_\theta^i$ of Eq. (2.7). Only one of the $N$ receivers (observers, for instance) can be synchronized according to the value of $m_k$ which is assumed to be constant within the interval of time $[jK_0, (j+1)K_0 - 1]$. A simple logical decoder enables to retrieve the original information when analyzing the residuals $r_k^i$, where

$$r_k^i = y_k - \tilde{h}_\theta^i(\hat{x}_k).$$

When multi-valued information is considered [PM01], the number of receivers increases and a sophisticated logical mechanism, located after the bank of receivers, is required.
Regarding a noisy context, the modulation technique is appealing because it benefits from some immunity properties. In a noise-free context though, it is much less attractive because it suffers from the fact that each switch of $m_k$ causes a transient in the synchronization process. That motivates the requirement that $m_k$ must be constant within an interval of time. Unfortunately, that prevents high throughput transmissions. To the lack of efficiency adds to the mounting number of receivers when $N$ becomes larger.

#### 2.3.2.2 Parameter modulation

Basically, there are two kinds of parameter modulation : the discrete and the continuous one. The setup corresponding to a discrete parameter modulation [UOL+93] [HPM93] is depicted in FIG. 2.5a. In such a case, a parameter depending on the input $m_k$ denoted $\lambda(m_k)$ (different from the secret key $\theta$) of a single chaotic system, takes values according to a prescribed rule over a finite set $\{\lambda^1, \ldots, \lambda^N\}$ : one has $\lambda(m_k) = \lambda^i$. For binary messages, the parameter of the transmitter only takes two distinct values $\lambda^1$, $\lambda^2$. During the interval of time $[jK_0, (j+1)K_0 - 1]$, $m_k$ is assumed to be constant and the chaotic signal $y_k$ is conveyed through the channel :

$$\left\{ \begin{array}{lcl} x_{k+1} & = & f_\theta^{\lambda(m_k)}(x_k), \\ y_k & = & h_\theta^{\lambda(m_k)}(x_k). \end{array} \right. \tag{2.8}$$

The receiver part can consist of a bank of $N$ receivers, usually observers, each of them being coupled in a unidirectional way with the transmitter through $y_k$ :

$$\left\{ \begin{array}{lcl} \hat{x}_{k+1} & = & \tilde{f}_\theta^{\lambda^i}(\hat{x}_k, y_k), \\ \hat{y}_k & = & \tilde{h}_\theta^{\lambda^i}(\hat{x}_k). \end{array} \right. \tag{2.9}$$

Only one observer, set with the same value $\lambda^i$ of the transmitter which has actually delivered the sequence $\{y_k\}_{jK_0}^{(j+1)K_0-1}$, can be synchronized according to (2.2) or (2.3) (where $\hat{x}_0$ must be replaced by $\hat{x}_{jK_0}$) within the time interval $[jK_0, (j+1)K_0 - 1]$ . Thus, again, a simple logical decoder permits to retrieve the original information when analyzing the residuals

$$r_k^i = y_k - \tilde{h}_\theta^{\lambda^i}(\hat{x}_k).$$

For the continuous modulation (FIG. 2.5b), the information $m_k$ takes values over an uncountable set. Consequently, an infinite number of units at the receiver side would be required. As a matter of fact, for the recovering of $\lambda(m_k)$ and then of $m_k$, we usually resort to adaptive techniques and identification procedures [FM97][CHR00][HM97][AMB04]. The estimation $\hat{\lambda}_k$ must achieve $\hat{\lambda}_k = \lambda(m_k)$ after a transient as short as possible.

For both discrete and continuous modulation, the function delivering $\lambda(m_k)$ must be bijective so that $m_k$ can be recovered in a unique way.

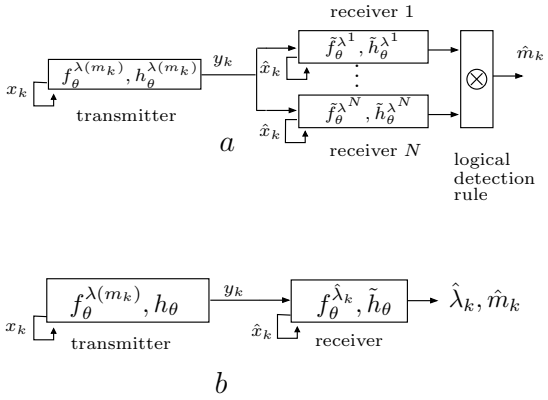


FIGURE 2.5 – Parameter modulation

Nevertheless, for the parameter modulation as with chaotic switching, the information must be constant during a prescribed interval of time (or at least slowly time-varying in a bounded range) to cope with the transients induced by the adaptation of the identification process. As with chaotic switching, this technique severely limits high throughput purposes and, therefore, it does not seem very appealing for encryption.

### 2.3.3 Two-channel transmission

For a two-channel transmission (FIG. 2.6), a first channel is used to convey the output $y_k$ of an autonomous chaotic system with dynamic $f_\theta$ and output function $h_\theta$. Besides, a function $\nu_e$, depending on a time-varying quantity, say, the state vector $x_k$ of the chaotic system, encodes the information $m_k$ and delivers $u_k = \nu_e(x_k, m_k)$. Then, the signal $u_k$ is transmitted via a second channel. The set of equations governing the transmitter is

$$\begin{cases} x_{k+1} = f_\theta(x_k), \\ y_k = h_\theta(x_k), \\ u_k = \nu_e(x_k, m_k). \end{cases} \tag{2.10}$$

At the receiver end, since the chaotic signal $y_k$ is information-free (and so not disturbed), a perfect synchronization fulfilling (2.2) or (2.3) can be achieved by resorting to an observer. As a consequence, the information $m_k$ can be correctly recovered by :

$$\begin{cases} \hat{x}_{k+1} = \tilde{f}_\theta(\hat{x}_k, y_k), \\ \hat{y}_k = \tilde{h}_\theta(\hat{x}_k), \\ \hat{m}_k = \nu_d(\hat{x}_k, u_k). \end{cases} \tag{2.11}$$

The decoding function $\nu_d$ is defined by

$$\hat{m}_k = \nu_d(\hat{x}_k, u_k) = m_k \text{ when } \hat{x}_k = x_k. \tag{2.12}$$

This technique has been proposed, for example, in [MM98][ZP02]. The advantage lies in that, unlike modulation-based approaches, $m_k$ is allowed to switch every discrete times $k$ without inducing synchronization transients for each symbol. The recovering is wrong only for a finite number of first symbols of the message. On the other hand, a transmission involving two channels may be unsatisfactory for throughput purposes.

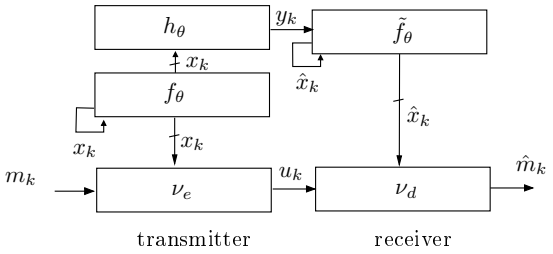


FIGURE 2.6 – Two-channel transmission

### 2.3.4 Message-embedding

First of all, the reader is cautioned that different but equivalent terminologies can be encountered in the literature referring to the same technique : embedding [KYP00][MD04], non autonomous modulation [Yan04] or direct chaotic modulation [Has98]. The reasons for this diversity are the following. At the transmitter part, the information $m_k \in \mathbb{R}$ is directly injected (or, as it is also usually said, embedded) in a chaotic dynamics $f_\theta$. The resulting system turns into a non autonomous one since the information acts as an exogenous input. The system involves a state vector $x_k \in \mathbb{R}^n$. Injecting $m_k$ into the dynamics can be considered as a "modulation" of the phase space. Only the output $y_k$ of the system is transmitted.

A message-embedded cryptosystem (depicted on FIG. 2.7) obeys :

$$\begin{cases} x_{k+1} = f_\theta(x_k, m_k), \\ y_k = h_\theta(x_k, m_k). \end{cases} \tag{2.13}$$

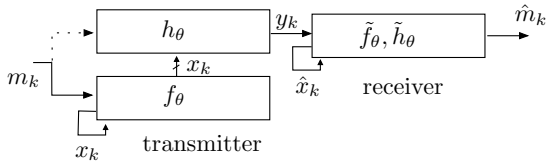


FIGURE 2.7 – Message-embedding. $m_k$ is embedded into $f_\theta$ and $h_\theta$ if the inherent delay is 0 or $m_k$ is only embedded into $f_\theta$ if the inherent delay is strictly greater than 0

For recovering the message at the receiver side, two mechanisms have been proposed in the literature : the inverse system approach [UMW96] and the unknown input observer (UIO) approach [ET01] [BDR02] [MD04] [MD03] [MD06] [BBBBT04]. As a matter of fact, UIO is nothing else but a left inverse system slightly modified by adding some extra terms, for the sake of robustness in noisy environments. The generic equations governing an inverse system or an UIO for (2.13) are :

$$\begin{cases} \hat{x}_{k+r+1} = \tilde{f}_\theta(\hat{x}_{k+r}, y_k, \ldots, y_{k+r}), \\ \hat{m}_{k+r} = \tilde{h}_\theta(\hat{x}_{k+r}, y_k, \ldots, y_{k+r}), \end{cases} \tag{2.14}$$

with $g_\theta$ such that

$$\hat{m}_{k+r} = \tilde{h}_\theta(\hat{x}_{k+r}, y_k, \ldots, y_{k+r}) = m_k \quad \text{when} \quad \hat{x}_{k+r} = x_k. \tag{2.15}$$

The index $r$ corresponds to the inherent delay of (2.13) and must be introduced, in particular, for the sake of causality.

This technique follows the same spirit as the observer-based techniques required for the cryptosystems described in Subsect. 2.3.1, 2.3.2 and 2.3.3. Nevertheless, a major difference lies in that the receiver, unlike a mere observer, must reconstruct the state $x_k$ to guarantee the synchronization without the knowledge of $m_k$.

Let $M$ be a constant matrix of appropriate dimension, and $U$ a non empty set of initial conditions. There are again two main concepts :

**Definition 12** *Unknown Input Asymptotical synchronization :*

$$\lim_{k\to\infty} \|Mx_k - \hat{x}_{k+r}\| = 0 \quad \forall \hat{x}_0 \in U \quad \textit{and} \quad \forall m_k \tag{2.16}$$

**Definition 13** *Unknown Input Finite Time synchronization :*

$$\exists k_f < \infty, \quad \|Mx_k - \hat{x}_{k+r}\| = 0 \quad \forall \hat{x}_0 \in U, \forall k \geq k_f \quad \textit{and} \quad \forall m_k. \tag{2.17}$$

**Remark 4** *Similarly to the mere synchronization, in practice, if a digital implementation is carried out, because of the finite accuracy of any computers, the error of an unknown input asymptotical synchronization can be considered to be zero after a finite transient time.*

The message-embedded approach is very appealing because not only, similarly to the two-channel approach, the recovering can be achieved without any assumption on the rate of variation of $m_k$ but also (unlike in a two-channel transmission), only a single channel is required.

## 2.4 Example of the message-embedding

The objective of this example is to illustrate the message embedding scheme described in Subsection 2.3.4.

**Transmitter**
We consider the switched linear dynamical SISO system (1.6) as the transmitter. The number of mode is $J = 2$ with the corresponding switching function $\sigma : k \in \mathbb{N} \to \{1, 2\}$ :

$$\sigma(k) = \begin{cases} 1 & \text{if } x_k^{(1)} > 0 \\ 2 & \text{if } x_k^{(1)} \leq 0 \end{cases}$$

All the matrices $A_{\sigma(k)}, B_{\sigma(k)}, C_{\sigma(k)}$ and $D_{\sigma(k)}$ are given as :

$$A_1 = \begin{pmatrix} -1.7 & 0.5 & 1 \\ 0.5 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad A_2 = \begin{pmatrix} 1.7 & 0.5 & 1 \\ 0.5 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.18}$$

$$B_1 = B_2 = B = [0.005\ 0.002\ 1]^T \text{ and } C_1 = C_2 = C = [1\ 1\ 1]$$

Actually, the dynamics of (2.18) is nothing but the Lozi map rewritten in a three-dimensional space to cope with the bias term which is incorporated in the dynamics $x_{k+1}^{(3)} = x_k^{(3)} = 1$.

The dynamical system (1.6) is used as the transmitter for encrypting a 24-bits colored image depicted in FIG. 2.8. Each 8 bits numerical value of the input $m_k \in \{0, \ldots, 255\}$ is extracted from a three dimensional array of integers which encodes the respectively red, green and blue layer of the image.
The random-look output signal $y_k$ is depicted in FIG. 2.9 and the resulting encrypted image is depicted in FIG. 2.10.

**Receiver**
In order to design the receiver, it is necessary to find out the inherent delay. To this end, we must compute the matrices $M_{\sigma(k)}^i$ as in (1.7). We have :

$$\begin{aligned} M_{\sigma(k)}^0 &= D_{\sigma(k)} = 0 \\ M_{\sigma(k)}^1 &= \begin{pmatrix} D_{\sigma(k)} & 0 \\ C_{\sigma(k+1)}B_{\sigma(k)} & D_{\sigma(k+1)} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1.007 & 0 \end{pmatrix} \end{aligned}$$

FIGURE 2.8 – The original image

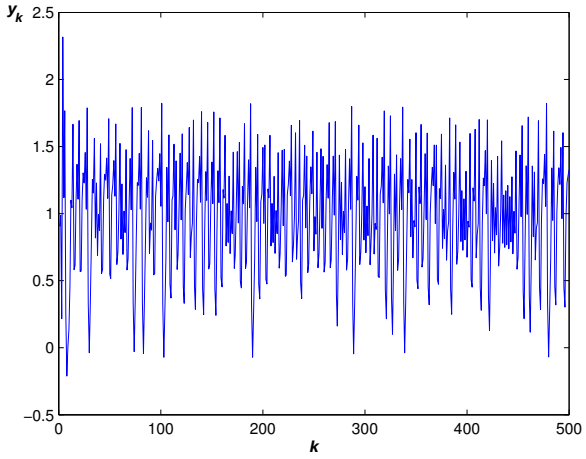


FIGURE 2.9 – A part of output signal $y_k$

and

$$rank\ M^1_{\sigma(k)} - rank\ M^0_{\sigma(k+1)} = 1$$

Hence, the inherent delay is $r = 1$ according to Theorem 1.

We can thereby suggest the inverse system (1.13) (actually the inverse system (2.14) particularized for switched linear systems) for recovering the image from the output sequence. We recall (see Subsection 1.2.2 in Chapter 1) that the error of reconstruction $\epsilon_k = x_k - \hat{x}_{k+r}$ fulfills

$$\epsilon_{k+1} = P^r_{\sigma(k)}\epsilon_k$$

To check whether the error state reconstruction is at least asymptotically stable (which would correspond to the relation (2.16) in the definition of an unknown

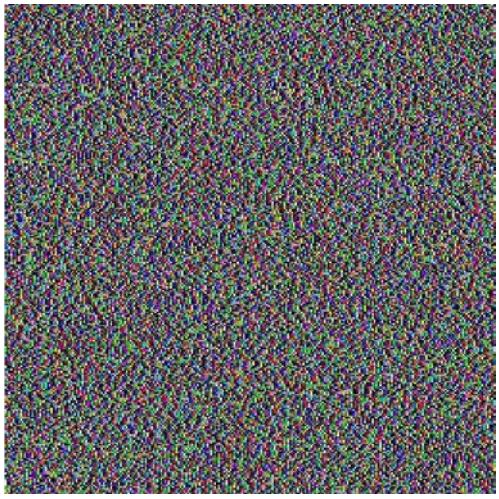

FIGURE 2.10 – Encrypted image

input asymptotical synchronization), we can test the quadratic and the polyquadratic stability (see Subsection 1.2.4 in Chapter 1 by substituting $A_{\sigma(k)}$ by $P^r_{\sigma(k)}$). From this perspective, we can resort to the freeware LMISOL [3]. The following results have been obtained.

– The error of synchronization is quadratically stable since we can find a common symmetric positive matrix

$$P = \begin{pmatrix} 1.1351 & 0.0615 & -0.2094 \\ 0.3039 & 0.9047 & 0.2559 \\ 0.0763 & 0.7164 & 0.6319 \end{pmatrix}$$

which verifies Theorem 4

– The error of synchronization is poly-quadratically stable since we can find matrices

$$S_1 = \begin{pmatrix} 0.9712 & 0.0347 & 0.0013 \\ 0.0347 & 0.8804 & 0.0209 \\ 0.0013 & 0.0209 & 0.9591 \end{pmatrix} \quad , \quad S_2 = \begin{pmatrix} 0.7328 & 0.3031 & 0.0085 \\ 0.3031 & 0.6506 & 0.0285 \\ 0.0085 & 0.0285 & 0.7456 \end{pmatrix}$$

$$G_1 = \begin{pmatrix} 0.9712 & 0.0347 & 0.0013 \\ 0.0347 & 0.8804 & 0.0209 \\ 0.0013 & 0.0209 & 0.9591 \end{pmatrix} \quad , \quad G_2 = \begin{pmatrix} 0.7328 & 0.3031 & 0.0085 \\ 0.3031 & 0.6506 & 0.0285 \\ 0.0085 & 0.0285 & 0.7456 \end{pmatrix}$$

which verify Theorem 5.

3. available at http ://www.dt.fee.unicamp.br/ mauricio/lmisol10.html

The error of synchronization is depicted in FIG. 2.11. We can easily realize that the Remark 4 applies. Indeed, after a finite transient time, because of the digitalization in the computation, the error reaches strictly zero and the image is properly recovered after a finite transient time (see FIG. 2.12).

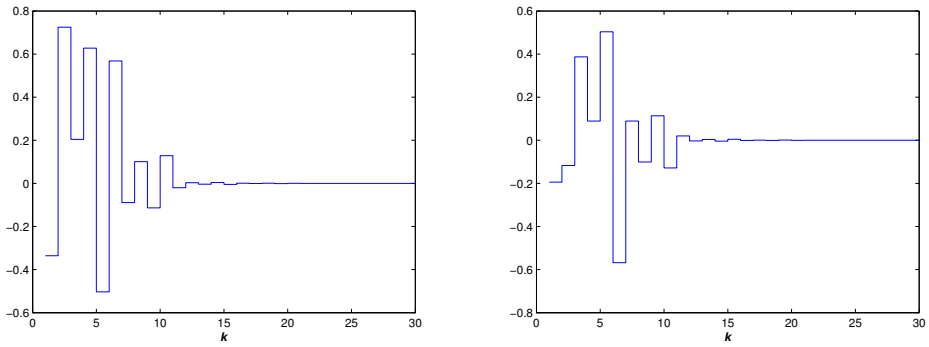


FIGURE 2.11 – Error state reconstruction $\epsilon_k^{(1)}$ (left) and $\epsilon_k^{(2)}$ (right).

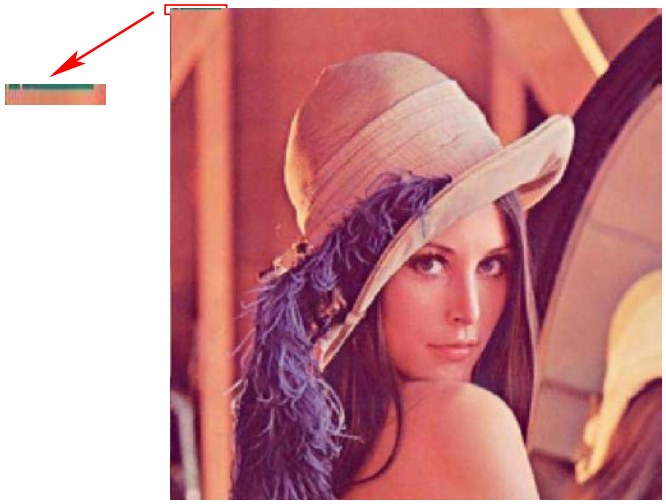

FIGURE 2.12 – Recovered image. Upper left : errors due to the synchronization transient

## 2.5 Conclusion

This chapter has recalled the most popular techniques for concealing information using chaos. Several limitations related to the performances concerning encryption/decryption speed, throughput, complexity of the receiver have been pointed out. The additive masking, which considers the information as a disturbance, is not able to reconstruct it accurately. The modulation method depends on the rate variation of the information and the structure of the receiver is likely

to be complex. The two-channel transmission suffers from throughput problem as it does require two channels, one for synchronization and another one for conveying the encrypted information. The message-embedding scheme appears to be the most attractive one. The receiver is unique and does not depend on the rate variation of the information to be encrypted.

However, if a digital application is sought (hardware implementation in e.g. FPGA or DSP), the data to be encrypted are either intrinsically digital or digitalized and so lie in a finite set. It is clear that resorting to a map which takes value in dense set (chaotic map and the underlying set of real values for example) will cause the output (the corresponding encrypted information) to also take value in a dense set. When implemented in a finite state machine, the output will be automatically quantized, the transient time before convergence will be finite, but clearly, this is a poor solution regarding the throughput. Undoubtedly, resorting to a map which directly takes value in a finite set whose range is identical to the one of the data would be a better solution. The next chapter is mainly dedicated to this important aspect.

# Chapter 3

# Message-embedding over a finite field

It has been shown in Chapter 2 that various cryptosystems, corresponding to different ways of hiding a message, have drawn the attention of the researchers over the years. The most important schemes are additive masking, chaotic switching, discrete or continuous parameter modulation, two-channel transmission, and message-embedding. The message-embedding appears to be very attractive as the synchronization between the transmitter and the receiver can be guaranteed without any restriction on the rate of variation of the message to be encrypted. However, in the context of digital applications, it has been stressed in Chapter 2 that resorting to maps which take value in a finite set whose range is identical to the one of the digital data would be a better solution.

The aim of this chapter is to revisit the message-embedding scheme where the dynamics is generated by switched linear systems over a finite field $\mathbb{F}_p$. SISO dynamical systems will be considered because they bring out some advantages. The switched linear system (transmitter part) has a simple inverse system (receiver part) which can be easily derived from the left invertibility and flatness condition. Furthermore, the resulting schemes are very convenient for implementation using hardware description languages (eg. VHDL). Considering a finite set, more precisely a finite field $\mathbb{F}_p$ instead of the dense set $\mathbb{R}$ implies many consequences :

- The "chaos" terminology does no longer make sense and must be replaced by the appropriate one "complex dynamics".

- The control-theoretical properties given in Chapter 1 need to be reconsidered in this new context. Indeed, we are only interested in the SISO case and we are able to derive a simple but effective inverse mechanism that fulfills the requirements related to computational speed and throughput.

- The parameters of a message-embedding scheme play the role of secret key. In our context, the attack consists in recovering the parameters through ac-

cessible sequences of inputs (the message) and the outputs (the encrypted information). That requires an identification algorithm. Classical identification of switched linear systems is no longer valid over finite fields and need to be adapted.

The outline of this chapter is as follows : Section 3.1 recalls the algebra which gives the background on computation over a finite field. Section 3.2 is dedicated to the design of the message-embedding scheme over a finite field. Switched linear systems and their control-theoretical properties are revisited in this new context. Identification of switched linear systems over a finite field is addressed in Section 3.3. We end up this chapter by a conclusion.

## 3.1 Recall on algebra

Before proceeding any further, we must recall the definition of a finite field and a polynomial ring. (see Appendix B for more details on algebra).

**Finite Field** :

Given any integer $a$ and a positive integer $p$, we define and denote $a \ (mod \ p)$ the division of $a$ by $p$ that leaves the remainder between 0 and $p-1$. We call two integers $a$ and $b$ to be congruent modulo $p$ if $a \ (mod \ p) = b \ (mod \ p)$ and we express such a congruence by $a = b \ (mod \ p)$. We denote $\mathbb{F}_p$ the set $\mathbb{F}_p = \{0, 1, \cdots, p-1\}$, that is the set of remainders in arithmetic modulo $p$. When $p$ is a prime number, $\mathbb{F}_p$ is a field.

Indeed, $\mathbb{F}_p$ is a ring, that is a set together with two laws of composition (two mappings $\mathbb{F}_p \times \mathbb{F}_p \mapsto \mathbb{F}_p$), namely the addition (denoted $+$) and the multiplication (denoted $\cdot$ or without any symbols) modulo $p$. The addition is associative and commutative and has a unit element denoted 0 (for every element $x \in \mathbb{F}_p$, the relation $x + 0 = 0 + x = x$ applies) and has an inverse (for every element $x \in \mathbb{F}_p$, there exists an element $y \in \mathbb{F}_p$ such that $x + y = y + x = 0$). The multiplication is associative and has a unit element denoted 1 (for every element $x \in \mathbb{F}_p$, the relation $x \cdot 1 = 1 \cdot x = x$ applies). Besides, distributivity of the addition over the multiplication applies (for all $x, y, z \in \mathbb{F}_p$ one has $(x+y)z = xz+yz$.

Furthermore, $\mathbb{F}_p$ is a division ring that is a ring such that $1 \neq 0$, and such that every non-zero element is invertible (for every element $x \in \mathbb{F}_p$ there exists an element $y \in \mathbb{F}_p$ such that $x \cdot y = y \cdot x = 1$). The existence of an inverse for every non-zero elements is guaranteed by the fact that $p$ is prime.

Finally, $\mathbb{F}_p$ is a field because it is a commutative division ring (that is the multiplication is commutative).

**Polynomial ring** :
A polynomial ring denoted $\mathbb{F}_p[z_k]$ or $\mathbb{F}_p[z_k^{(1)}, \ldots, z_k^{(i)}, \ldots, z_k^{(n)}]$ is a ring of whose elements are polynomials. The indeterminates are the vector components $z_k^{(i)}$ and the coefficients lie in $\mathbb{F}_p$.

All along this chapter, the addition and the multiplication are performed modulo $p$ and for shortness, $(mod\ p)$ will be omitted.

## 3.2 Design

### 3.2.1 Transmitter part

As the transmitter part of the message-embedding cryptosystem, we consider the SISO switched linear dynamical system :

$$\begin{cases} x_{k+1} &= A_{\sigma(k)}x_k + B_{\sigma(k)}m_k \\ y_k &= C_{\sigma(k)}x_k + D_{\sigma(k)}m_k \end{cases} \tag{3.1}$$

where $m_k \in \mathbb{F}_p$, $y_k \in \mathbb{F}_p$ and $x_k \in \mathbb{F}_p^n$. The switching function $\sigma$

$$\sigma : k \in \mathbb{N} \mapsto j = \sigma(k) \in \{1, \ldots, J\}$$

is arbitrary, in particular no dwell time is assumed. All the matrices, namely $A_{\sigma(k)} \in \mathbb{F}_p^{n\times n}$, $B_{\sigma(k)} \in \mathbb{F}_p^{n\times 1}$, $C_{\sigma(k)} \in \mathbb{F}_p^{1\times n}$ and $D_{\sigma(k)} \in \mathbb{F}_p$ belong to the respective finite sets $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ and $(D_j)_{1\leq j\leq J}$. At a given time $k$, the index $j$ corresponds to the mode of the system given by the switching function $\sigma$.

### 3.2.2 Receiver part

It has been stressed in Chapter 2 that the receiver part of a message-embedded cryptosystem must have the form of an inverse system and that the existence is guaranteed through the left inversion property. Algebraic characterizations had been provided in Chapter 1. It is shown in this subsection that, because the transmitter is SISO the expression of the inverse system can be simplified. Finally, some specific algorithms for inverting over $\mathbb{F}_p$ are provided herein.

#### 3.2.2.1 Left invertibility

We provide a condition for checking for the left invertibility of (3.1) which is simpler than the rank condition (1.12)

To this end, we first find out the expression of $y_{k+i}$ by iterating (3.1).

$$y_{k+i} = C_{\sigma(k+i)} A_{\sigma(k)}^{\sigma(k+i-1)} x_k + \sum_{j=0}^{j=i} \mathcal{T}_{\sigma(k)}^{i,j} m_{k+j} \tag{3.2}$$

with

$$\mathcal{T}_{\sigma(k)}^{i,j} = C_{\sigma(k+i)} A_{\sigma(k+j+1)}^{\sigma(k+i-1)} B_{\sigma(k+j)} \text{ if } j \leq i-1, \quad \mathcal{T}_{\sigma(k)}^{i,i} = D_{\sigma(k+i)} \tag{3.3}$$

and with the direct transition matrix defined as :

$$A_{\sigma(k_0)}^{\sigma(k_1)} = \begin{cases} A_{\sigma(k_1)} A_{\sigma(k_1-1)} \dots A_{\sigma(k_0)} & \text{if } k_1 \geq k_0 \\ \mathbf{1}_n & \text{if } k_1 < k_0 \end{cases}$$

Let us define a quantity $r$ as follows :

- $r = 0$ if $\mathcal{T}_{\sigma(k)}^{0,0} \neq 0$ for all $k$
- the least integer such that for all $k$

$$\begin{array}{l} \mathcal{T}_{\sigma(k)}^{i,j} = 0 \text{ for } \; i = 0, \dots, r-1 \; \text{ and } \; j = 0, \dots, i \\ \mathcal{T}_{\sigma(k)}^{r,0} \neq 0 \end{array} \tag{3.4}$$

If $r$ is finite ($r < \infty$), the output at time $k + r$ is given as :

$$y_{k+r} = C_{\sigma(k+r)} A_{\sigma(k)}^{\sigma(k+r-1)} x_k + \mathcal{T}_{\sigma(k)}^{r,0} m_k \tag{3.5}$$

and the input $m_k$ can be deduced in a unique way as :

$$m_k = (\mathcal{T}_{\sigma(k)}^{r,0})^{-1} (y_{k+r} - C_{\sigma(k+r)} A_{\sigma(k)}^{\sigma(k+r-1)} x_k) \tag{3.6}$$

According to the Definition 4, $r$ is actually the inherent delay of (3.1) and (3.5) defines the function $h_{x_k}$ (see Eq. (1.4) in Chapter 1).

**Remark 5** *Actually the inherent delay $r$ coincide in this case with the relative degree $R$ according to the Definition 3 (see Chapter 1))*

We can derive the following Proposition :

**Proposition 1** *The system (3.1) is left invertible if it has a finite and constant inherent delay $r$ (or equivalently relative degree $R = r$)*

**Remark 6** *The existence of the inverse of $\mathcal{T}_{\sigma(k)}^{r,0}$ is guaranteed since, by definition (see Eq. (3.4)), it is always different from zero and we recall that every non-zero element has an inverse in $\mathbb{F}_p$. On the other hand, if $p$ would not have been prime, the relative degree would no longer have coincided with the inherent delay. Indeed, $\mathbb{F}_p$ would have been reduced to a ring for which not every non-zero element have an inverse, the condition on the existence of an inverse turning into $gcd(\mathcal{T}_{\sigma(k)}^{r,0}, p) = 1$. Hence the explicit dependence (3.5) of $y_{k+r}$ on $m_k$ would not have been necessarily induced that (3.6) holds.*

#### 3.2.2.2 Left inversion

We are now concerned with a recursive left inversion of (3.1) achieving the recovery of $m_k$ from $y_k$ without any knowledge of $x_k$. Owing to the fact that a transmitter in a form of a SISO system is considered, we show that the expression of the inverse system can be simpler than (1.13).

Let us define the inverse transition matrix as

$$P^{\sigma(k_1)}_{\sigma(k_0)} = \begin{cases} P^r_{\sigma(k_1)} P^r_{\sigma(k_1-1)} \cdots P^r_{\sigma(k_0)} & \text{if } k_1 \geq k_0 \\ \mathbf{1}_n & \text{if } k_1 < k_0 \end{cases}$$

with

$$P^r_{\sigma(k)} = A_{\sigma(k)} - B_{\sigma(k)} (\mathcal{T}^{r,0}_{\sigma(k)})^{-1} C_{\sigma(k+r)} A^{\sigma(k+r-1)}_{\sigma(k)} \tag{3.7}$$

**Proposition 2** *Assume that (3.1) is left invertible and has inherent delay $r$. The following dynamical system is a $r-$delayed inverter for (3.1).*

$$\begin{cases} \hat{x}_{k+r+1} &= P^r_{\sigma(k)} \hat{x}_{k+r} + B_{\sigma(k)} (\mathcal{T}^{r,0}_{\sigma(k)})^{-1} y_{k+r} \\ \hat{m}_{k+r} &= -(\mathcal{T}^{r,0}_{\sigma(k)})^{-1} C_{\sigma(k+r)} A^{\sigma(k+r-1)}_{\sigma(k)} \hat{x}_{k+r} \\ & \quad +(\mathcal{T}^{r,0}_{\sigma(k)})^{-1} y_{k+r} \end{cases} \tag{3.8}$$

**Proof 1** *On one hand, substituting (3.5) into (3.8) yields :*

$$\begin{array}{rcl} \hat{x}_{k+r+1} &=& P^r_{\sigma(k)} \hat{x}_{k+r} + B_{\sigma(k)} (\mathcal{T}^{r,0}_{\sigma(k)})^{-1} C_{\sigma(k+r)} A^{\sigma(k+r-1)}_{\sigma(k)} x_k \\ & & + B_{\sigma(k)} (\mathcal{T}^{r,0}_{\sigma(k)})^{-1} \mathcal{T}^{r,0}_{\sigma(k)} m_k \end{array} \tag{3.9}$$

*Taking into account (3.7) and noticing that $(\mathcal{T}^{r,0}_{\sigma(k)})^{-1} \mathcal{T}^{r,0}_{\sigma(k)} = 1$, $\epsilon_k = x_k - \hat{x}_{k+r}$ fulfills the recursion :*

$$\begin{array}{rcl} \epsilon_{k+1} &=& (A_{\sigma(k)} - B_{\sigma(k)} (\mathcal{T}^{r,0}_{\sigma(k)})^{-1} C_{\sigma(k+r)} A^{\sigma(k+r-1)}_{\sigma(k)}) \epsilon_k \\ & & = P^r_{\sigma(k)} \epsilon_k \end{array} \tag{3.10}$$

*On the other hand, from the expression (3.6) of $m_k$ and the expression of $\hat{m}_{k+r}$ in (3.8), we get that :*

$$m_k - \hat{m}_{k+r} = -(\mathcal{T}^{r,0}_{\sigma(k)})^{-1} C_{\sigma(k+r)} A^{\sigma(k+r-1)}_{\sigma(k)} (x_k - \hat{x}_{k+r}) \tag{3.11}$$

**Remark 7** *Clearly, defining a stable $r-$delayed inverter in terms of uniformly asymptotical stability of the system $\epsilon_{k+1} = P^r_{\sigma(k)} \epsilon_k$ makes sense in $\mathbb{R}$ but does no longer make sense in the finite field like $\mathbb{F}_p$. Only the finite time stability still holds. It is defined as follows.*

Let $U_p$ be either a subset of $\mathbb{F}_p$ or $\mathbb{F}_p$ itself.

**Definition 14** *The system (3.10) is finite time stable if*

$$\exists k_f < \infty, \quad \|\epsilon_k\| = 0 \ \ \forall \epsilon_0 \in U_p \ \ \textit{and} \ \ \forall k \geq k_f. \tag{3.12}$$

Finite Time Stability induces Unknown Input Finite Time synchronization with (3.1) (see (2.17) in Chapter 2).

We are now in position of characterizing the finite time stability.

**Proposition 3** *The system (3.10) is finite time stable whenever there exists an integer $K < \infty$ such that for all $k \geq 0$*

$$P_{\sigma(k)}^{\sigma(k+K-1)} = \mathbf{0} \tag{3.13}$$

**Remark 8** *We must notice that the condition (3.13) is nothing but the condition (1.17) characterizing flatness. We can thereby conclude that, over $\mathbb{F}_p$, the only transmitters for which an acceptable (because finite time stable) receiver is guaranteed are the flat systems. This is an important difference compared to $\mathbb{R}$.*

We can obtain the expression of $x_k$. Indeed, if (3.1) is left invertible and has inherent delay $r$, (3.8) exists. Iterating (3.8) $l-1$ times yields :

$$\begin{array}{rcl} \hat{x}_{k+r+l} & = & P_{\sigma(k)}^{\sigma(k+l-1)} \hat{x}_{k+r} \\ & & + \sum_{i=0}^{l-1} P_{\sigma(k+i+1)}^{\sigma(k+l-1)} B_{\sigma(k+i)} \mathcal{T}_{\sigma(k+i)}^{r,0} y_{k+i+r} \end{array} \tag{3.14}$$

If (3.13) is fulfilled, (3.14) turns into

$$\hat{x}_{k+r+K} = \sum_{i=0}^{K-1} P_{\sigma(k+i+1)}^{\sigma(k+K-1)} B_{\sigma(k+i)} \mathcal{T}_{\sigma(k+i)}^{r,0} y_{k+i+r} \tag{3.15}$$

revealing that $\hat{x}_{k+r+K}$ is independent of $\hat{x}_{k+r}$. In particular, (3.15) holds for $\hat{x}_{k_0+r} = x_{k_0}$ for all $k_0 \geq 0$, that is for $\epsilon_{k_0} = 0$ with $k_0 \geq 0$. We infer that $\epsilon_k = 0$ for all $k \geq k_0$ and thus $\hat{x}_{k+r+K} = x_{k+K}$ for all $k \geq 0$. Therefore, after performing the change of variable $k \rightarrow k-K$, we obtain an explicit form for $x_k$ :

$$x_k = \sum_{i=0}^{K-1} P_{\sigma(k+i+1-K)}^{\sigma(k-1)} B_{\sigma(k+i-K)} \mathcal{T}_{\sigma(k+i-K)}^{r,0} y_{k+i+r-K} \tag{3.16}$$

#### 3.2.2.3 Methodology for inverting over a finite field

To complete the design of the receiver, we must provide a methodology for computing the (multiplicative) inverse of $\mathcal{T}_{\sigma(k)}^{r,0} \neq 0$ which is an element in $\mathbb{F}_p$. Regarding this problem, there exist two approaches : i) Greatest common divisor approach, ii) Fermat's little theorem approach.

**Greatest Common Divisor-based approach**
The first one is based on the computation of the *gcd* (greatest common divisor) of $\mathcal{T}_{\sigma(k)}^{r,0}$ and $p$. Indeed, since $p$ is prime and $\mathcal{T}_{\sigma(k)}^{r,0} < p$, we have :

$$gcd(\mathcal{T}^{r,0}_{\sigma(k)}, p) = 1$$

From the Bezout's lemma, there exists two integers $\alpha, \beta$, such that :

$$\alpha\mathcal{T}^{r,0}_{\sigma(k)} + \beta p = 1$$

This yields :

$$\alpha\mathcal{T}^{r,0}_{\sigma(k)} = 1 \ (mod\ p)$$

Consequently, we obtain :

$$(\mathcal{T}^{r,0}_{\sigma(k)})^{-1} = \alpha \ (mod\ p)$$

The corresponding algorithm for computing the integers $\alpha, \beta$ are the Extended Euclidean Algorithm [MOV96] and the binary algorithm [Knu98].

Let $\mathcal{T}(i,j)$, $\mathcal{T}(i,:)$ denote respectively the component at $i$-th row and $j$-th column and the $i$-th row of the matrix $\mathcal{T}$ defined as

$$\mathcal{T} = \begin{pmatrix} \mathcal{T}^{r,0}_{\sigma(k)} & 1 & 0 \\ p & 0 & 1 \end{pmatrix}$$

Let $Quot(\mathcal{T}^{r,0}_{\sigma(k)}, p)$ denote the quotient of the division $\frac{\mathcal{T}^{r,0}_{\sigma(k)}}{p}$. The Extended Euclidean algorithm for completing the multiplicative inverse of $\mathcal{T}^{r,0}_{\sigma(k)}$ is described below.

**Algorithm 1** Extended Euclidean Algorithm

**Input :** $\mathcal{T}^{r,0}_{\sigma(k)}$, $p$
**Output :** $inva \in \mathbb{F}_p$ % multiplicative inverse of $\mathcal{T}^{r,0}_{\sigma(k)}$

Construct the matrix $\mathcal{T} = \begin{pmatrix} \mathcal{T}^{r,0}_{\sigma(k)} & 1 & 0 \\ p & 0 & 1 \end{pmatrix}$
Set the variable $Continue$ = true

**while** $Continue$ **do**

$tmpRow = \mathcal{T}(2,:)$
$\mathcal{T}(2,:) = \mathcal{T}(1,:) + \mathcal{T}(2,:) \cdot (-Quot(\mathcal{T}(1,1), \mathcal{T}(2,1)))$
$\mathcal{T}(1,:) = tmpRow$
**if** $\mathcal{T}(2,1) == 0$ **then** $Continue$ = false
**end if**

**end while**

$inva = \mathcal{T}(1,2)$ % multiplicative inverse of $\mathcal{T}^{r,0}_{\sigma(k)}$

**Fermat's Little Theorem-based approach**
The second approach is based on the Fermat's Little Theorem. For any given non-zero $\mathcal{T}^{r,0}_{\sigma(k)} \in \mathbb{F}_p$, we have :

$$(\mathcal{T}^{r,0}_{\sigma(k)})^{p-1} \ (mod\ p) = 1$$

This yields :

$$(\mathcal{T}^{r,0}_{\sigma(k)})^{p-2}\mathcal{T}^{r,0}_{\sigma(k)} \ (mod\ p) = 1$$

As a result, we obtain :

$$(\mathcal{T}^{r,0}_{\sigma(k)})^{-1} = (\mathcal{T}^{r,0}_{\sigma(k)})^{p-2} \ (mod\ p)$$

For this approach, the multiplicative inverse is obtained by computing the exponent which is nothing but the multiple multiplications over the finite field. The most effective algorithm for performing this task is the Montgomery algorithm [Mon85].

## 3.3 Identification over finite fields

Let $\theta$ be a parameter vector consisting of a subset of entries of $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ and $(D_j)_{1\leq j\leq J}$ in the state space model (3.1). As these parameters are expected to act as the secret key, we must present an identification procedure of $\theta$.

### 3.3.1 General principle

Since the unauthorized party has in general no access to the internal state (state vector $x_k$), the general principle can be based on the corresponding input/output model of (3.1). It has been highlighted (see Remark 8) that only flat switched systems are acceptable candidates to act as a transmitter. When the system (3.1) is flat, its input/output model can be obtained in a systematic and convenient way. Indeed, if (3.1) is flat with flat output $y_k$, the state vector $x_k$ obeys (3.16). Substituting the expression (3.16) of $x_k$ into (3.5) yields directly the input/output relation :

$$\begin{array}{l} y_{k+r} = C_{\sigma(k+r)}A^{\sigma(k+r-1)}_{\sigma(k)}\cdot \\ (\sum_{i=0}^{K-1} P^{\sigma(k-1)}_{\sigma(k+i+1-K)}B_{\sigma(k+i-K)}\mathcal{T}^{r,0}_{\sigma(k+i-K)}y_{k+i+r-K}) + \mathcal{T}^{r,0}_{\sigma(k)}m_k \end{array} \tag{3.17}$$

Let $\{\sigma_1\}^{k+r-1}_{k+r-K}, \ldots, \{\sigma_N\}^{k+r-1}_{k+r-K}$ the $N$ possible mode sequences $\{\sigma(k+r-K), \ldots, \sigma(k+r-1)\}$ over the interval of time $[k+r-K, k+r-1]$. The number $N$ of all possible mode sequences is finite since the number $J$ of modes of (3.1) is. These mode sequences will be respectively denoted for short $\sigma_1, \ldots, \sigma_N$ in the sequel. Thus, for $t = 1, \ldots, N$, the input/output relation (3.17) can be rewritten as

$$y_{k+r} = \sum_{j=0}^{K-1} a_j(\sigma_t)y_{k+j+r-K} + c(\sigma_t)m_k \tag{3.18}$$

where $c(\sigma_t)$ and the $a_j(\sigma_t)$'s ($j = 0, \ldots, K-1$) are coefficients depending, in different ways according to the sequence $\sigma_t$, on the entries of the matrices $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ and $(D_j)_{1\leq j\leq J}$ of (3.1)

**Proposition 4** *The maximum number $N = N_{I/O}$ of input/ouput relations regardless of the number $J$ of modes is $N_{I/O} = p^{K+1}$*

**Proof 2** *The proof is an immediate consequence of the two following claims. The input/output relation (3.18) involves $K+1$ coefficients. Each of them takes value in the set $\mathbb{F}_p$ which is of finite cardinality $p$.*

Based on (3.18), two different procedures according to the accessibility of $\sigma_t$ can be suggested for the identification of $c(\sigma_t)$ and the $a_j(\sigma_t)$'s.

$\sigma_t$ is accessible
Since for each $\sigma_t$, the parameters $c(\sigma_t)$ and the $a_j(\sigma_t)$'s appear in a linear fashion in the input/output relation (3.18), the identification is easy. Indeed, for a given mode sequence $\sigma_t$, the identification can be performed by iterating the relation (3.18) until a set of linear independent equations is obtained and can be solved.

$\sigma_t$ is not accessible
The previous procedure does no longer work. On the other hand, it can be inspired from the method proposed in [VMS03] for switched ARX systems over $\mathbb{R}$. The method is adapted to our context and described below.

Each input/output relation (3.18) can be rewritten for $t = 1, \ldots, N$ as :

$$z_k^T b_t = 0 \tag{3.19}$$

with

- $z_k = [y_{k+r}, y_{k+r-1}, \cdots, y_{k+r-K}, m_k]^T \in \mathbb{F}_p^{K+2}$
- $b_t = [1, -a_0(\sigma_t), \ldots, -a_{K-1}(\sigma_t), -c(\sigma_t)]^T \in \mathbb{F}_p^{K+2}$

$z_k$ is the *regressor vector* while $b_t$ is the *parameter vector* corresponding to the mode sequence $\sigma_t$.

We can thereby define $N$ hyperplanes $S_t$, $t = 1 \ldots, N$

$$S_t = \{z_k : z_k^T b_t = 0\}$$

The following equation applies regardless of the switching sequences :

$$p_N(z_k) = \prod_{t=1}^{N} (z_k^T b_t) = \nu_N(z_k)^T h_N = 0 \tag{3.20}$$

It is called *Hybrid Decoupling Constraint* equation and $p_N$ is the *Hybrid Decoupling Constraint Polynomial*.

**Remark 9** *The first component $h_N^{(1)}$ of $h_N$ equals 1*

Since the multiplication is closed in the ring $\mathbb{F}_p[z_k]$, the product $p_N(z_k)$ is also in $\mathbb{F}_p[z_k]$.

$h_N \in \mathbb{F}^{M_N}$ is the coefficient of the *Hybrid Decoupling Polynomial* and $\nu_N : z_k \in \mathbb{F}_p^{K+2} \mapsto \xi_k \in \mathbb{F}_p^{M_N}$ is a *Veronese map* of degree $N$, the components of $\xi_k$ corresponding to all the $M_N$ monomials (product of the components $z_k^{(i)}$ of $z_k$) sorted in the degree-lexicographic order [4]

The quantity $M_N$ depends on $K$ and is given by

$$M_N(K) = \left( \begin{array}{c} N+K+1 \\ N \end{array} \right) = \frac{(N+K+1)!}{N!(K+1)!} \tag{3.21}$$

For shortness, $M_N(K)$ will be sometimes merely written $M_N$ in the sequel.

For the identification of the $b_t$'s in (3.19), it is first required to compute the coefficients $h_N$ of (3.20). Then $b_t$ can be derived.

**Computing $h_N$**
Let $\mathcal{L}_N$ denote the embedded data matrix involving $N'$ mapped regressor vectors $z_k$ through $\nu_N$

$$\mathcal{L}_N = \left[ \begin{array}{c} \nu_N(z_{k_1}) \\ \nu_N(z_{k_2}) \\ \ldots \\ \nu_N(z_{k_{N'}}) \end{array} \right]^T \in \mathbb{F}_p^{N' \times M_N} \tag{3.22}$$

The following relation applies :

$$\mathcal{L}_N h_N = \mathbf{0} \tag{3.23}$$

$N'$ is an integer large enough such that the $\nu_N(z_{k_i})$'s $(i = 1, \ldots, N')$ can span a $M_N - 1$ dimensional vector space, i.e

$$rank(\mathcal{L}_N) = M_N - 1 \tag{3.24}$$

The lower bound of $N'$ is obviously $M_N - 1$. If (3.24) is fulfilled, the coefficient $h_N$ can be retrieved by

$$h_N = Ker(\mathcal{L}_N) \tag{3.25}$$

4. A *lexicographic order* is a ranking according to the names of the variables and their iterates such that :
   - $z_k^{(i)} < z_{k+l}^{(i)}, \forall l \in \mathbb{N}$,
   - $z_m^{(i)} < z_l^{(j)} \Rightarrow z_{m+t}^{(i)} < z_{l+t}^{(j)}, \forall m \in \mathbb{N}, \forall l \in \mathbb{N}, \forall t \in \mathbb{N}$,
   - $z_k^{(i)} < z_k^{(j)} \Rightarrow (z_k^{(i)})^\alpha < (z_k^{(j)})^\beta, \forall \alpha \in \mathbb{N}, \forall \beta \in \mathbb{N}$

To find out the kernel in (3.25), there exist two methods : i) Gaussian elimination over $\mathbb{F}_p$, ii) Gaussian-Bareiss elimination technique (see Appendix C for the details). The first method replaces the division operation by multiplying the (multiplicative) inverse over the finite field $\mathbb{F}_p$. To determine the multiplicative inverse, we refer to the subsection 3.2.2.3. The second one performs the elementary row operation in a specific way so that the result of division operation is still an integer number.

**Computing $b_t$**
Let us recall the following definition :

**Definition 15** *[Lan02] A derivation $D$ on the field $\mathbb{F}_p$ is a mapping $D: \mathbb{F}_p \mapsto \mathbb{F}_p$ which is linear and satisfies the ordinary rule for derivatives, ie., for every element $x, y$ in $\mathbb{F}_p$,*

$$D(x+y) = D(x) + D(y) \text{ and } D(x.y) = xD(y) + yD(x)$$

As a result, the derivative $Dp_N(z_k)$ of $p_N(z_k)$ in (3.20) is also in the polynomial ring $\mathbb{F}_p[z_k]$ and reads :

$$Dp_N(z_k) = \frac{\partial p_N(z_k)}{\partial z_k} = \frac{\partial}{\partial z_k} \prod_{t=1}^{N} ({z_k}^T b_t) = \sum_{t=1}^{N} b_t \prod_{l \neq t}^{N} ({z_k}^T b_l) \tag{3.26}$$

We rewrite (3.26) as :

$$Dp_N(z_k) = b_t \prod_{l \neq t}^{N} ({z_k}^T b_l) + \sum_{i \neq t}^{N} b_i \prod_{j \neq i}^{N} ({z_k}^T b_j) \tag{3.27}$$

Now, consider an arbitrary vector $w_t \in \mathbb{F}_p^{K+2}$, such that, $w_t^T b_t = 0$. Replacing $w_t$ $(t = 1, \ldots, N)$ into (3.27) yields :

$$Dp_N(w_t) = b_t \prod_{l \neq t}^{N} (w_t^T b_l) = b_t.c \tag{3.28}$$

where $c$ is a scalar. Thus, the parameter vectors $b_t$'s $(t = 1, .., N)$ is obtained by normalizing (3.28)

To determine the $N$ distinct points $w_t$ that lie on the $N$ hyperplanes $S_t$, the following algebraic procedure can be carried out.

Consider a parametrized random line with direction $v$ and a base point $w_0$ :

$$\mathcal{D} : \mu v + w_0 \ \forall \mu \in \mathbb{Z}$$

The line $\mathcal{D}$ intersects with all the hyperplanes at $N$ distinct intersections under the condition that it is not parallel with any of the hyperplanes. The three dimensional case is illustrated in FIG. 3.1. In others words, the equation of degree $N$

$$p_N(\mu v + w_0) = 0 \tag{3.29}$$

has $N$ distinct integer roots $\{\mu_t\}_{t=1}^{N}$ under the constraint $p_N(v) \neq 0$ (or equivalently $v \notin S_t$). Therefore, the intersection of this line and all of the hyperplanes are given by :

$$w_t = \mu_t v + w_0 \ \forall t \in \{1, .., N\}$$

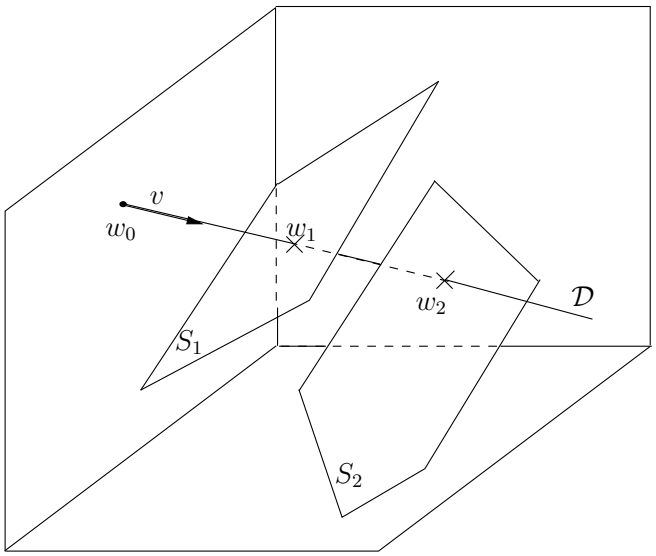


FIGURE 3.1 – The intersection of a random line $\mathcal{D}$ and two planes $S_1$, $S_2$ in a three-dimensional space

**Remark 10** *An alternative approach can be also suggested. Since $w_t$ belongs to a finite field, an exhaustive search for finding out $\mu_t$ could be effective as well.*

### 3.3.2 Unicity

When the switching sequences are unknown, the identification procedure requires to compute the solution of (3.23), that is finding out the kernel $h_N$ of $\mathcal{L}_N$. The one-dimensionality of the solution is guaranteed by the rank condition (3.24) recalled below :

$$rank(\mathcal{L}_N) = M_N - 1$$

Actually, whenever the one-dimensionality of the solution $h_N$ is guaranteed, its unicity is as well. Indeed, let us recall (see Remark 9) that a normalization must

be performed to ensure that the first component $h_N^{(1)}$ of $h_N$ equals 1.

When working over $\mathbb{R}$, the assumption that the mapped regressor vectors $\nu_N(z_{k_i})$ are *sufficiently exciting* is known as the PE condition. Over a finite field like $\mathbb{F}_p$, the PE conditions make no longer sense. Indeed, the number of possible regressors $z_{ki}$ is finite over $\mathbb{F}_p$. The objective of this subsection is to provide necessary conditions under which the rank condition can be fulfilled.

**Proposition 5** *The maximum number $N' = N'_{max}$ of regressors $z_{ki}$ that (3.1) can generate, regardless of the number $J$ of modes, is $N'_{max} = p^{K+1}$*

**Proof 3** *The number of components of the regressor vector $z_k$ is $K+2$. On the other hand, regarding (3.17), the component $y_{k+r}$ is linearly congruent to the other ones $y_{k+r-1}, \ldots, y_{k+r-K}, m_k$. These $K+1$ components take value in the set $\mathbb{F}_p$ which is of finite cardinality $p$. That completes the proof.*

Besides, the Veronese map in (3.20)

$$\nu_N : z_k \in \mathbb{F}_p^{K+2} \mapsto \xi_k \in \mathbb{F}_p^{M_N}$$

is surjective over the finite field $\mathbb{F}_p$. Thus, the cardinality of the sets $\{z_k\}$ and $\{\xi_k\}$ fulfills :

$$card\left(\{\xi_k\}\right) \leq card\left(\{z_k\}\right) = N_{reg} = p^{(K+1)}$$

This implies :

$$rank(\mathcal{L}_N) \leq N'_{max} = p^{(K+1)} \tag{3.30}$$

Based on the relations (3.24) and (3.30), it can be inferred that a necessary condition for the one-dimensionality (and so unicity) of the kernel $h_N$ is that the triplet $(p, K, N)$ is such that :

$$N'_{max} = p^{(K+1)} \geq M_N(K) - 1 \tag{3.31}$$

The following proposition applies :

**Proposition 6** *For all pairs $(p, K_c)$ with $p \geq 2$, there exists an integer $N \in [1, N_{I/O}]$ so that :*

$$N'_{max} = p^{(K+1)} \geq M_N(K) - 1$$

*for $K \geq K_c$*

**Proof 4** *We recall the expression (3.21) of $M_N(K)$ :*

$$M_N(K) = \begin{pmatrix} N+K+1 \\ N \end{pmatrix} = \frac{(N+K+1)!}{N!(K+1)!}$$

*On one hand, since* $M_1(K) - 1 = K + 1$*, it is clear that, for all* $p \geq 2$

$$p^{(K+1)} > M_1(K) - 1 \tag{3.32}$$

*On the other hand, let us first show that*

$$p^{(K+1)} < M_{N_{I/O}}(K) - 1 \tag{3.33}$$

*It is clear that for all* $p \geq 2$ *we have :*

$$\begin{array}{rl} p^{K+1} + 2 > & 2 \\ \vdots & \vdots \\ p^{K+1} + K + 1 > & K + 1 \end{array}$$

*Multiplying all the terms in the lefthand side and the righthand side yields :*

$$\prod_{i=2}^{K+1} (p^{K+1} + i) > (K + 1)!$$

*Multiplying both sides by* $(p^{(K+1)} + 1)!$ *yields :*

$$\begin{array}{l} (p^{(K+1)} + 1)! \prod_{i=2}^{K+1} (p^{K+1} + i) > (p^{(K+1)} + 1)!(K + 1)! \\ (p^{(K+1)} + K + 1)! > (p^{(K+1)} + 1)!(K + 1)! \end{array}$$

*Dividing both sides by* $(p^{(K+1)})!(K + 1)!$ *yields*

$$\frac{(p^{(K+1)} + K + 1)!}{(p^{(K+1)})!(K + 1)!} > (p^{(K+1)} + 1)$$

*and so*

$$\frac{(p^{(K+1)} + K + 1)!}{(p^{(K+1)})!(K + 1)!} - 1 > p^{(K+1)}$$

*Yet, from (3.21) and taking into account that* $N_{I/O} = p^{K+1}$*, the following equality applies*

$$M_{N_{I/O}}(K) = \frac{(p^{(K+1)} + K + 1)!}{(p^{(K+1)})!(K + 1)!}$$

*which proves (3.33).*

*Finally, it is easy to see that the functions* $K \to p^{K+1}$ *and* $K \to M_N(K) - 1$ *for any* $N$ *are monotonic increasing functions of* $K$*.*

*As a result, for any prescribed pairs* $(p, K_c)$ *with* $p \geq 2$*, there exists an integer* $N \in [1, N_{I/O}]$ *so that the functions* $K \to p^{K+1}$ *and* $K \to M_N(K) - 1$ *intersect each other, and so there exists* $N$ *such that* $p^{(K+1)} > M_N(K) - 1$ *for* $K \geq K_c$*.*

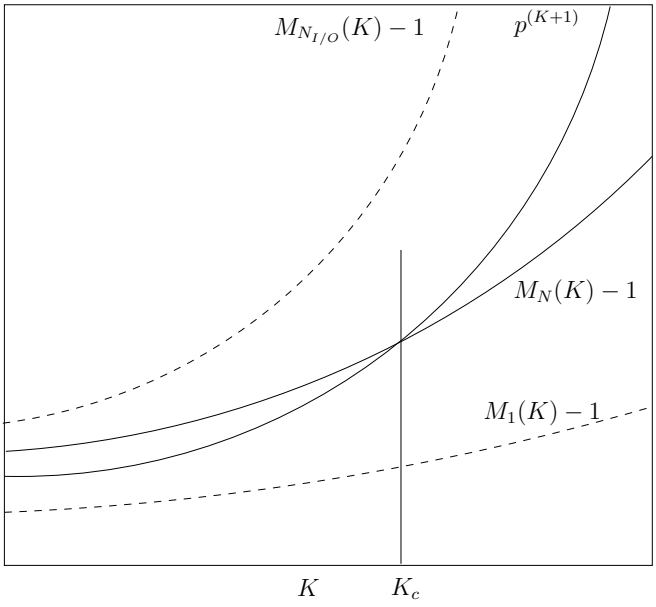


FIGURE 3.2 – Graphical interpretation of the Proposition 6

A graphical interpretation of the Proposition 6 can be made and is illustrated in FIG. 3.2. For a prescribed $p$, all the pairs $(K, N)$ with $K < K_c$ for which the curve $p^{K+1}$ is lower than the curve $M_N(K) - 1$ prevent the rank condition (3.30) to be fulfilled and so the unicity of $h_N$.

**Remark 11** *The converse is not true. Even if (3.31) holds, the rank condition (3.30) may not be fulfilled. Indeed, owing to the dynamics of the system, the number $N'$ of independent regressors $z_{ki}$ may be lower than the maximum number $N'_{max}$*

**Remark 12** *Proposition 6 can be used in order to choose an appropriate number of modes $J$ since the number of input/ouput relations $N$ is related to $J$.*

## 3.4 Examples

The purpose of this section is to illustrate through different examples the identification procedure and the impact of the choices of the triplets $\{p, N, K\}$.

### 3.4.1 Example 1

Consider a one-dimensional switched dynamical system over the finite field $\mathbb{F}_2$ ($p = 2$) of the form (3.1) with $A_{\sigma(k)} = q_{\sigma(k)} \in \{0, 1\}$, $B_{\sigma(k)} = C_{\sigma(k)} = 1$ and $D_{\sigma(k)} = 0$.

**Inherent delay**
According to (3.4), the inherent delay is $r = 1$ since $CB \neq 0$.

**Flatness**
According to (3.13), the system is flat since for all $k$, $P_{\sigma(k)}^{\sigma(k+K-1)}$ with $K = 1$ verifies $P_{\sigma(k)}^{\sigma(k+K-1)} = P_{\sigma(k)} = 0$.

**Identification based on the input/output relation**
In view of (3.17), the corresponding input/ouput model reads :

$$y_{k+1} = q_{\sigma(k)} y_k + m_k$$

The regressor vector is given by $z_k = [y_{k+1}, y_k, m_k]^T$ and, according to the Proposition 5, the maximum number of regressors is $N'_{max} = p^{K+1} = 2^{1+1} = 4$.

According to the Proposition 4, the maximum number of input/output relations are $N_{I/O} = p^{K+1} = 4$ but actually, here there only exists two input/output relations according to the value of $q_{\sigma(k)}$. Hence, $N = 2$.

For $N = 2$ and $K = 1$, $M_N(K) - 1 = 5$. Thus, the necessary rank condition (3.30) is not fulfilled and the kernel will not be unique.

Applying the Gaussian-Bareiss algorithm yields precisely four possible vectors $h_N$ :

$$h_N \in ( \begin{array}{l} [1,1,1,0,0,0]^T, [1,1,0,0,1,1]^T, \\ [1,0,0,1,0,1]^T, [1,0,1,1,1,0]^T ) \end{array}$$

To each kernel vector $h_N$, we can find the corresponding parameter vector $b_t$. Only the kernel vector $[1,1,0,0,1,1]^T$ gives the right solution for the $b_t$'s : $b_1 = [1,1,1]^T$ and $b_2 = [1,0,1]^T$ .

**Remark 13** *The maximum number of regressors is $N'_{max} = p^{K+1} = 2^{1+1} = 4$ but actually, only two independent regressors are obtained. That explains the reason why there are four distinct solutions in $h_N$ : $M_N(K) - 2 = 6 - 2 = 4$*

### 3.4.2 Example 2

Consider a three-dimensional switched dynamical system over the finite field $\mathbb{F}_2$ ($p = 2$) of the form (3.1) with

$$A_{\sigma(k)} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & q_{\sigma(k)} & 1 \end{pmatrix} \ , \ B_{\sigma(k)} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \ , \ C_{\sigma(k)} = [1 \ \ 0 \ \ 0] \ , \ D_{\sigma(k)} = 0$$

and $q_{\sigma(k)} \in \{0, 1\}$

**Inherent delay**
According to (3.4), the inherent delay is $r = 3$ since basic manipulation yields $\mathcal{T}^{3,0}_{\sigma(k)} = C_{\sigma(k+3)} A^{\sigma(k+2)}_{\sigma(k+1)} B_{\sigma(k)} = 1 \neq 0$.

**Flatness**
According to (3.13), the system is flat. Indeed,

$$\begin{aligned} P^3_{\sigma(k)} &= A_{\sigma(k)} - B_{\sigma(k)} (\mathcal{T}^{3,0}_{\sigma(k)})^{-1} C_{\sigma(k+3)} A^{\sigma(k+2)}_{\sigma(k)} \\ &= \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & q_{\sigma(k)} & 1 \end{pmatrix} - \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & q_{\sigma(k)} & 1 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \end{aligned}$$

and for all $k$, $P^{\sigma(k+K-1)}_{\sigma(k)}$ with $K = 3$ fulfills

$$P^{\sigma(k+K-1)}_{\sigma(k)} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}^3 = \mathbf{0}$$

**Identification based on the input/output relation**
In view of (3.17), the corresponding input/ouput model reads :

$$y_{k+3} = y_{k+2} + q_{\sigma(k)} y_{k+1} + y_k + m_k$$

The regressor vector is given by $z_k = [y_{k+3}, y_{k+2}, y_{k+1}, y_k, m_k]^T \in \mathbb{F}_2^5$ and, according to the Proposition 5, the maximum number of regressors is $N'_{max} = p^{K+1} = 2^{3+1} = 16$.

According to the Proposition 4, the maximum number of input/output relations are $N_{I/O} = p^{K+1} = 16$ but actually, there only exists two input/output relations according to the value of $q_{\sigma(k)}$. Hence, $N = 2$.

For $N = 2$ and $K = 3$, $M_N(K) - 1 = 14$. Thus, the necessary rank condition (3.30) is fulfilled. Applying the Gaussian-Bareiss algorithm yields a unique vector $h_N$ :

$$h_N = [1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0]^T$$

We find the parameter vectors $b_1 = [1, 1, 1, 1, 1]^T$ and $b_2 = [1, 1, 0, 1, 0]^T$ from the kernel vector $h_N$.

### 3.4.3 Example 3

Consider a one-dimensional switched dynamical system over the finite field $\mathbb{F}_{251}$ ($p = 251$) of the form (3.1) with $A_{\sigma(k)} = q_{\sigma(k)} \in \mathbb{F}_{251}$ , $B_{\sigma(k)} = 5$ , $C_{\sigma(k)} = 1$ , $D_{\sigma(k)} = 0$. The switching function $\sigma(k)$ is not accessible and defined by :

$$\sigma : k \in \mathbb{N} \mapsto \sigma(k) = j \in \{1, 2\}$$

and $q_{\sigma(k)} = \{q_1, q_2\} = \{38, 213\}$

**Inherent delay**
According to (3.4), the inherent delay is $r = 1$ since $\mathcal{T}^{1,0}_{\sigma(k)} = C_{\sigma(k+1)}B_{\sigma(k)} = 5 \neq 0$.

**Flatness**
According to (3.13), the system is flat since for all $k$, $P^{\sigma(k+K-1)}_{\sigma(k)}$ with $K = 1$ verifies $P^{\sigma(k+K-1)}_{\sigma(k)} = P_{\sigma(k)} = 0$.

**Multiplicative inverse of $\mathcal{T}^{1,0}_{\sigma(k)}$**
The receiver governed by Eq. (3.8) aims at recovering the input $m_k$ and the multiplicative inverse of $\mathcal{T}^{1,0}_{\sigma(k)} = 5$ over the finite field $\mathbb{Z}_{251}$ must be computed. The greatest common divisor approach and the Extended Euclidean Algorithm 1 are used. The following successive matrices $\mathcal{T}$ are obtained :

$$\mathcal{T} = \begin{pmatrix} 5 & 1 & 0 \\ 251 & 0 & 1 \end{pmatrix}$$

$$\text{Step 1} : \mathcal{T} = \begin{pmatrix} 251 & 0 & 1 \\ 5 & 1 & 0 \end{pmatrix}$$

$$\text{Step 2} : \mathcal{T} = \begin{pmatrix} 5 & 1 & 0 \\ 1 & 201 & 1 \end{pmatrix}$$

$$\text{Step 3} : \mathcal{T} = \begin{pmatrix} 1 & 201 & 1 \\ 0 & 251 & 246 \end{pmatrix}$$

At step 3, the Algorithm 1 stops since $\mathcal{T}(2, 1) = 0$ and we derive the multiplicative inverse of 5 over $\mathbb{Z}_{251}$ : $inva = \mathcal{T}(1, 2) = 201$.

**Identification based on the input/output relation**
In view of (3.17), the input/ouput model reads :

$$y_{k+1} = q_{\sigma(k)}y_k + 5m_k \tag{3.34}$$

We have two parameter vectors $b_1 = [1, -q_1, -5]^T$ and $b_2 = [1, -q_2, -5]^T$. Since $213 = -38$ ($mod$ 251) and $246 = -5$ ($mod$ 251) one has $b_1 = [1, 213, 246]^T$ and $b_2 = [1, 38, 246]^T$.

The regressor vector is given by $z_k = [y_{k+1}, y_k, m_k]^T$ and, according to the Proposition 5, the maximum number of regressors is $N'_{max} = p^{K+1} = 251^{1+1} = 63001$.

According to the Proposition 4, the maximum number of input/output relations are $N_{I/O} = p^{K+1} = 63001$ but actually, there only exists two input/output relations according to the value of $q_{\sigma(k)}$. Hence, $N = 2$.

For $N = 2$ and $K = 1$, $M_N(K) - 1 = 5$. Thus, the necessary rank condition (3.30) is fulfilled. Consequently, the vector coefficient $h_N$ can be unique.

**Computing $h_N$**
Starting with a random input sequences

$$\{m_k\}_{k=1}^{9} = \{144, 231, 93, 162, 11, 122, 2, 176, 127\}$$

the corresponding output sequence is given by

$$\{y_k\}_{k=1}^{9} = \{0, 218, 150, 36, 195, 186, 68, 187, 205\}$$

Thus, we obtain $N' = 8$ regressor vectors :

$$\begin{aligned}
z_{k1} &= [218 \quad 0 \quad 144]^T \\
z_{k2} &= [150 \quad 218 \quad 231]^T \\
z_{k3} &= [36 \quad 150 \quad 93]^T \\
z_{k4} &= [195 \quad 36 \quad 162]^T \\
z_{k5} &= [186 \quad 195 \quad 11]^T \\
z_{k6} &= [68 \quad 186 \quad 122]^T \\
z_{k7} &= [187 \quad 68 \quad 2]^T \\
z_{k8} &= [205 \quad 187 \quad 176]^T
\end{aligned}$$

The embedded data matrix $\mathcal{L}_N$ involving $N' = 8$ mapped regressor vector through the Veronese map $\nu_N$ is given by (3.22) and numerically reads :

$$\mathcal{L}_N = \begin{pmatrix}
85 & 0 & 17 & 0 & 0 & 154 \\
161 & 70 & 12 & 85 & 158 & 149 \\
41 & 129 & 85 & 161 & 145 & 115 \\
124 & 243 & 215 & 41 & 59 & 140 \\
209 & 126 & 38 & 124 & 137 & 121 \\
106 & 98 & 13 & 209 & 102 & 75 \\
80 & 166 & 123 & 106 & 136 & 4 \\
108 & 183 & 187 & 80 & 31 & 103
\end{pmatrix}$$

Applying the Gaussian-Bareiss algorithm in the Appendix C with matrix $\mathcal{L}_N \in \mathbb{F}_{251}^{8\times 6}$ yields the upper-triangular form

$$\mathcal{L}_N = \begin{pmatrix}
85 & 0 & 17 & 0 & 0 & 154 \\
0 & 177 & 40 & 197 & 127 & 170 \\
0 & 0 & 211 & 138 & 215 & 204 \\
0 & 0 & 0 & 87 & 109 & 226 \\
0 & 0 & 0 & 0 & 175 & 0
\end{pmatrix}$$

and the kernel reads after normalization (see Remark 9)

$$h_N = [1,\ 0,\ 241,\ 62,\ 0,\ 25]^T$$

**Computing** $b_t$
First, we show how to compute $N = 2$ points $w_t$, such that, $w_t^T b_t = 0$. Consider a random line with a direction $v = [25, 181, 61]^T$ and a base point $w_0 = [42, 155, 208]^T$.

Solving (3.29) (or an exhaustive search according to the Remark 10) yields $t_1 = 59$, $t_2 = 197$ and two corresponding intersections $w_1 = [11, 41, 42]^T$, $w_2 = [198, 170, 177]^T$.

Finally, the parameter vectors $b_t$ according to (3.28) are given by :

$$b_1 = [1, 213, 246]^T$$
$$b_2 = [1,\ 38, 246]^T$$

We obtain exactly the two expected parameter vectors $b_t$.

## 3.5 Conclusion

This chapter has considered the message-embedding scheme over a finite field. We conclude that the dynamical system used in the transmitter part has to be flat. Moreover, it turns out that there are several advantages resorting to SISO systems. Indeed, in the case of MIMO switched linear systems, the inverse system (1.13) would be more complex than in the SISO case. It could be seen as a parallel algorithm for recovering simultaneously $m$ inputs. However, this parallelism would be ineffective. The computation of the pseudo-inverse over a finite field in (1.13) degrades the decryption speed comparing to the SISO case (compare with the Equation (3.8)).

Finally, for recovering the parameters which are expected to act as the secret key, an identification procedure has been proposed for switched linear systems over a finite field.

Keeping in mind these features, a validation of this kind of cryptosystems needs a further step which consists in a comparison with the conventional ciphers. The next chapter is devoted to this study.

# Chapter 4

# A connection with standard cryptography

In this chapter, we survey the different ciphers and the corresponding design methodologies encountered in standard cryptography. Next, we bring out a connection between chaos-based cryptosystems and standard ciphers. The investigation relies on structural consideration.

The outline of this Chapter is as follows : Section 4.1 recalls the background on standard encryption schemes with special emphasis on the so-called stream ciphers. Section 4.2 brings out the connection between the additive masking and the so-called Synchronous Stream Cipher on one hand, the message-embbeding and the Self-Synchronizing Stream Cipher on the other hand. A particularization for switched linear systems is made. Section 4.3 provides the state of the art in the design of Self-Synchronizing Stream Ciphers along with some examples. The distinct design methodologies are compared and it is proved why the message-embedding scheme involving flat dynamical systems appear as a new solution for the design of Self-Synchronizing Stream Ciphers. Section 4.4 deals with identifiability in connection with the concept of security.

## 4.1 Class of ciphers in standard cryptography

### 4.1.1 Generalities on cryptography

The considerable progress in communication technology during the last decades has led to an increasing need for security in information exchanges. In this context, cryptography plays a major role as information is mostly conveyed through public networks. The main objective of cryptography is, precisely, to conceal the content of messages transmitted through insecure channels, to unauthorized users or, in other words, to guarantee privacy and confidentiality in the communications.

Since the early 1960s, cryptography has no longer been restricted to military or governmental concerns, which has spurred an unprecedented development of it. At the same time, this development benefited very much from the advances in digital communication technology in form of new and efficient ways of designing encryption schemes. Modern cryptography originates in the works of Claude Shannon after World War II [Sha49].

In a general encryption mechanism, also called cryptosystem or cipher, we are given an alphabet $A$, that is, a finite set of basic elements named symbols. On the *transmitter* part, a plaintext (also called information or message) $m \in \mathcal{M}$ ($\mathcal{M}$ is called the message space) consisting of a string of symbols $m_k \in A$ is encrypted according to an encryption function $e$ which depends on the key $k^e \in \mathcal{K}$ ($\mathcal{K}$ is called the key space). The resulting ciphertext $c \in \mathcal{C}$ ($\mathcal{C}$ is called the ciphertext space), a string of symbols $c_k$ from an alphabet $B$ usually (and assumed hereafter) identical to $A$, is conveyed through a public channel to the *receiver*. At the receiver side, the ciphertext $c$ is decrypted according to a decryption function $d$ which depends on the key $k^d \in \mathcal{K}$. For a prescribed $k^e$, the function $e$ must be invertible.

Among a wide variety of cryptographic techniques, two major classes can be typically distinguished : *public-key* ciphers (or asymmetric-key ciphers) and *secret-key* ciphers (also called symmetric-key ciphers).

Public-key ciphers are largely based upon computationally very demanding mathematical problems, for instance, integer factorization into primes. The year 1976 is a milestone with the seminal paper of Diffie and Hellmann [WM76] that founded the public key cryptography. The year 1978 has been marked by the publication of RSA, the first full-fledged public-key algorithm. This discovery was important notably because it solved the key-exchange problem of symmetric cryptography. Actually, the key $k^e$ is public whereas $k^d$ is secret.

In symmetric encryption, the pair $(e, d)$ is such that the key $k^d$ can be easily recovered from $k^e$. Hence, not only $k^d$ must be kept secret but the key $k^e$ as well. It is customary that both keys are identical, that is $k^d = k^e$. There are two classes of symmetric-key encryption schemes which are commonly distinguished : block ciphers and stream ciphers.

A block cipher is an encryption scheme that breaks up the plaintext messages into strings (called blocks) of a fixed length over an alphabet and encrypts one block at a time. Block ciphers usually involve compositions of substitution and transposition operations. A key date in the recent history of cryptography is 1977, when the block cipher Data Encryption Standard (DES) was adopted by the U.S. National Bureau of Standards (now the National Institute of Standards and Technology - NIST), for encrypting unclassified information. DES is now in the process of being replaced by the Advanced Encryption Standard (AES), a new standard adopted by NIST in 2001.

Stream ciphers are mainly based on generators of complex sequences in the form of dynamical systems, which must be synchronized at the transmitter and receiver sides. We thereby realize why dynamical systems exhibiting complex dynamics, in particular chaotic, have a connection with cryptography. As we shall investigate this connection thoroughly, we must detail this class of ciphers.

### 4.1.2 Stream ciphers

In the case of stream ciphers, the encryption (*resp.* decryption) function $e$ (*resp.* $d$) can change for each symbol because it depends on a time-varying key $z_k$ (*resp.* $\hat{z}_k$) also called *running key*. The sequence $\{z_k\}$ (*resp.* $\{\hat{z}_k\}$) is called the *keystream*.
This being the case, stream ciphers are generally well appropriate and their use can even be compulsory when buffering is limited or when only one symbol can be processed at a time : the field of telecommunications often includes such constraints.

Stream ciphers require a keystream generator which is parametrized by the secret key $k^e = k^d = \theta$. It is usual that the plaintext $m_k$ and the ciphertext $c_k$ are binary words. If so, the most widely adopted function $e$ is the bitwise XOR operation and if the generator delivers a truly random keystream $\{z_k\}$ which is never used again, the encryption scheme is called *one-time pad* - the only cipher known to be unconditionally secure so far. However, in order to decrypt the ciphertext, the recipient party of a one-time pad encryption setup would have to know the random keystream and, thus, would require again a secure transmission of the key. Besides, for the *one-time pad* cipher, the key should be as long as the plaintext and would drastically increase the difficulty of the key distribution. As an alternative to such an ideal encryption scheme, one can resort to pseudo-random generators. Indeed, for such generators, the keystream is produced by a deterministic function (often involving feedback shift registers along with nonlinearities [Knu98]) while its statistical properties look random. There are two classes of stream ciphers, the difference lying in the way the keystream is generated : the synchronous stream ciphers and the Self-Synchronizing Stream Ciphers.

Synchronous Stream Ciphers (written hereafter SSC for short) admit the equations :

$$\begin{cases} q_k = \sigma^s(q_{k-1}) \\ z_k = s(q_k) \\ c_k = e(z_k, m_k) \end{cases} \tag{4.1}$$

$\sigma^s$ is the next-state transition function while $s$ acts as a filter and generates the keystream $\{z_k\}$.

Self-Synchronizing Stream Ciphers (written hereafter SSSC for short) admit the equations :

$$\begin{cases} z_k = \sigma_\theta^{ss}(c_{k-l-M}, \ldots, c_{k-l}) \\ c_k = e(z_k, m_k) \end{cases} \tag{4.2}$$

$\sigma_\theta^{ss}$ is the function that generates the keystream $\{z_k\}$. $l$ is a nonnegative integer standing for a possible delay. $\sigma_\theta^{ss}$ depends on past values of $c_k$. The number of past values is most often bounded and equals $M$, the *delay of memorization*.

Regardless the class of ciphers, synchronous or self-synchronizing, the ciphertext $c_k$ is worked out through an encryption function $e$ which must be invertible for any prescribed $z_k$. In the binary case, one has $A = B = \{0, 1\}$ and $e(z_k, m_k) = z_k \oplus m_k$ where $\oplus$ denotes the modulo 2 addition on the 2-element field. The decryption is performed through a function $d$ depending on the ciphertext $c_k$ and the running key $\hat{z}_k$ of the receiver's generator. Such a function must obey the rule :

$$\hat{m}_k := d(c_k, \hat{z}_k) = m_k \text{ if } \hat{z}_k = z_k \tag{4.3}$$

In the binary case, one has $d(\hat{z}_k, c_k) = \hat{z}_k \oplus c_k$

**Synchronization issues**
For stream ciphers, the generators at both sides have same generator function and synchronization of keystreams $\{z_k\}$ and $\{\hat{z}_k\}$ generated respectively at the transmitter and receiver sides is a condition for proper decryption.

For SSC, the generators are not coupled each other. Consequently, the only way to guarantee synchronization of the keystreams is to share the seed (the initial running key $z_0$). This being the case, the secret key $\theta$ is nothing but the seed $z_0$.

For SSSC, since the generator function $\sigma_\theta^{ss}$ shares, at the transmitter and receiver sides, the same quantities, namely the past ciphertexts, it is clear that the generators synchronize automatically after a finite transient time of length $M$. The secret key is some suitable (according to the security) parameters of the function $\sigma_\theta^{ss}$.

## 4.2 Connection between standard stream ciphers and chaotic cryptosystems

### 4.2.1 Additive masking vs synchronous stream ciphers

A natural connection can be made between additive masking and SSC. In fact,the transmitter of the respective schemes has exactly the same structure. The sequences $\{x_k\}$ for chaotic cryptosystems (*resp.* $\{z_k\}$ for SSC) are independent from the plaintext $m_k$ and the ciphertext $y_k$ (*resp.* $c_k$). The standard stream

ciphers involve pseudorandom generators over finite fields and require an initialization process at both ends to ensure synchronization. For additive masking, the generator is chaotic and synchronization is inevitably lost within a very short time window due to sensitivity to initial conditions. To handle such a problem, a controlled synchronization at the receiver part usually based on observers, is often suggested as mentioned in Section 2.3.1. The resulting cipher does no longer belong to the class of SSC. Besides, as pointed out in Chapter 2, the added information to be masked acts as a disturbance and prevents the control from guaranteeing an exact synchronization. This renders the additive masking not more appealing than a conventional SSC.

### 4.2.2 Message-embedding vs self-synchronizing stream ciphers

#### 4.2.2.1 General case

We first recall the general equations of the transmitter for the message-embedding (see Chapter 2).

$$\begin{cases} x_{k+1} = f_\theta(x_k, m_k), \\ y_k = h_\theta(x_k, m_k). \end{cases}$$

We examine two assumptions labelled $H1$ and $H2$.

$H1$ : the transmitter is left invertible with inherent delay $r$.
Hence, the map (see Chapter 1)

$$h_{x_k} : \begin{array}{ccc} A & \longrightarrow & A \\ m_k & \longmapsto & y_{k+r} = h^{(r)}(x_k, m_k) \end{array}$$

is well-defined and is a bijection.

$H2$ : the transmitter is flat with flat output $y_k$ and a flatness characteristic number $t_2 - t_1 + 1$.
Hence, the state vector $x_k$ obeys (see Eq. (5) in Chapter 1))

$$x_k \;=\; F(y_{k+t_1}, \cdots, y_{k+t_2})$$

The following Proposition brings out a connection between the message-embedding and an SSSC.

**Proposition 7** *If the system (2.13) fulfills the following assumptions $H1$ and $H2$*
*– it is left invertible with inherent delay $r$ ($H1$)*
*– it is flat with flat output $y_k$ and a flatness characteristic number $t_2 - t_1 + 1$ ($H2$)*
*then it is structurally equivalent to a self-synchronizing stream cipher of the form (4.2) with the correspondences (presented below for short by the symbol $\leftrightarrow$)*
*– a keystream generator (also named ciphering function) $\sigma_\theta^{ss} \leftrightarrow F$*

– *a running key* $z_k \leftrightarrow x_k$
– *a ciphertext* $c_{k+r} \leftrightarrow y_{k+r}$
– *a ciphering function* $e \leftrightarrow h^{(r)}$

Identification of the equations and properties derived from assumptions $H1$ and $H2$ with (4.2) gives the correspondence.

**Remark 14** *When the inherent delay $r$ is strictly greater than zero, there is a delay $r$ between the plaintext $m_k$ and the corresponding ciphertext $y_{k+r}$. It is similar to what typically happens when the output function of an SSC is pipelined (see the algorithm Moustique described in Subsection 4.3.2.2).*

The equivalent representation of the message-embedded cryptosystem is depicted on FIG. 4.1.

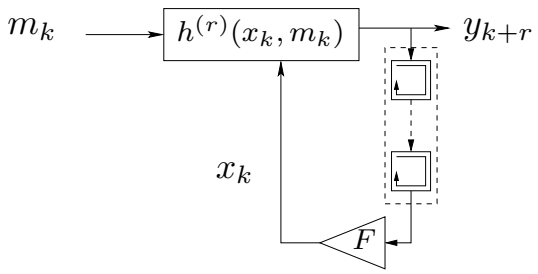


FIGURE 4.1 – Self-synchronizing Message Embedded Stream Cipher

#### 4.2.2.2 Particularization for switched linear systems

We turn back to the SISO switched linear dynamical system (3.1) of which equations are recalled below :

$$\begin{cases} x_{k+1} &= A_{\sigma(k)}x_k + B_{\sigma(k)}m_k \\ y_k &= C_{\sigma(k)}x_k + D_{\sigma(k)}m_k \end{cases}$$

The following conditions provide conditions under which (3.1) is structurally equivalent to a self-synchronizing stream cipher.

**Proposition 8** *(3.1) is structurally equivalent to a self-synchronizing stream cipher if :*
– *(3.1) has a left inherent delay $r$*
– *$y_k$ is a flat output*

**Remark 15** *Implicitly the switching rule $\sigma$ must be self-synchronizing itself*

**Proof 5** *If (3.1) has a left inherent delay $r$ and is flat, by virtue of (3.5) and (3.16) (see Chapter 3), the system (3.1) can be rewritten in the following equivalent form :*

$$\left\{ \begin{array}{lcl} x_k & = & \sum_{i=0}^{K-1} P_{\sigma(k+i+1-K)}^{\sigma(k-1)} B_{\sigma(k+i-K)} \mathcal{T}_{\sigma(k+i-K)}^{r,0} y_{k+i+r-K} \\ y_{k+r} & = & C_{\sigma(k+r)} A_{\sigma(k)}^{\sigma(k+r-1)} x_k + \mathcal{T}_{\sigma(k)}^{r,0} m_k \end{array} \right. \tag{4.4}$$

*and the result follows from the identification of (4.4) with (4.2), the correspondences being :*
– $y_k \leftrightarrow c_k$ *(ciphertext)*
– $x_k \leftrightarrow z_k$ *(keystream)*
– $(y_{k+r-K}, \ldots, y_{k+r-1}) \mapsto x_k \leftrightarrow \sigma_\theta^{ss}$ *(keystream generator)*
– $(x_k, m_k) \mapsto C_{\sigma(k+r)} A_{\sigma(k)}^{\sigma(k+r-1)} x_k + \mathcal{T}_{\sigma(k)}^{r,0} m_k \leftrightarrow e$ *(encryption function)*
– $r \leftrightarrow b_s$ *(delay)*

Actually, the model (4.2) of an SSSC is a conceptual model, called canonical representation, that can correspond to different architectures and that result from different design approaches. In the open literature, few designs methods have been proposed. They are detailed below in a way which highlights the central role played by dynamical systems and the reason why some concepts borrowed from control theory appear to be useful.

## 4.3 State of the art in the design of SSSC and examples

### 4.3.1 Block ciphers in CFB mode

This SSSC design approach resorts to a length $M$ shift register and a block cipher (DES for instance) both inserted in a closed-loop architecture. It is a very special mode of operation involving block ciphers naturally called Cipher Feed-Back (CFB) mode. The block cipher's input is the shift register state. Usually a limited number of the block cipher output bits are retained, the selection being performed through a so-called filter function denoted $h'$ on the FIG. 4.2. Such a configuration is often used in 1-bit CFB mode. In such a case, the encryption function $e$ is a XOR (modulo 2 addition over $\{0, 1\}$). The keystream generator $\sigma_\theta^{ss}$ of the corresponding canonical form (4.2) results from the composition of three functions : the state transition function of the shift register, the block cipher and the filter function $h'$.

This mode is quite inefficient in terms of encryption speed since one block cipher operation, and so multiple rounds, are required for enciphering a single plaintext $m_k$.

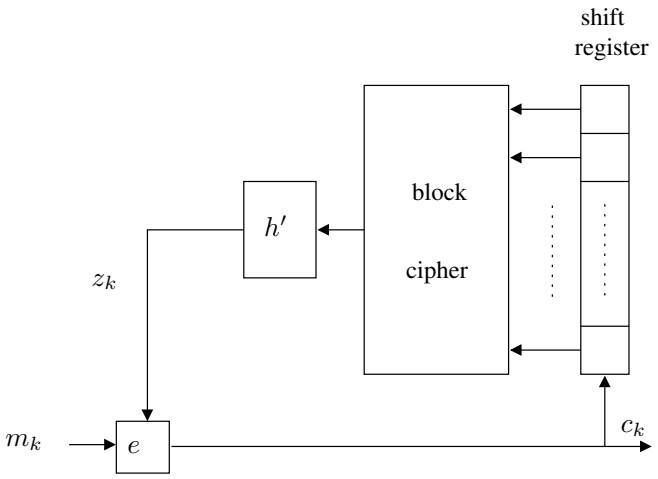


FIGURE 4.2 – Block cipher in CFB mode

### 4.3.2 Maurer's approach

In [Mau91], it is suggested an alternate design approach exclusively dedicated to SSSC. It includes two main ideas.

The first idea consists in replacing the shift register, the block cipher and the output bit filter function of the CFB mode architecture by an automaton. The automaton obeys the dynamics

$$\begin{cases} q_{k+1} &= g_\theta(q_k, c_k) \\ z_k &= h_\theta(q_k) \end{cases} \tag{4.5}$$

The function $g_\theta$ is the (next) state transition function while $h_\theta$ is the output function.

The automaton must have a finite input memory of size $M$ meaning that the state $q_k$ must be expressed by mean of a function $l_\theta$ which depends on a finite number of past ciphertexts $c_{k-i}$ :

$$q_k = l_\theta(c_{k-M}, \ldots, c_{k-1}) \tag{4.6}$$

Substituting the above expression of $q_k$ into the second equation of (4.5) gives the function $\sigma_\theta^{ss}$ of the canonical form (4.2). One has the following composition : $\sigma_\theta^{ss} = h_\theta \circ l_\theta$. According to the discussion of Subsect. 4.1.2 on synchronization issues, self-synchronization is guaranteed.

Let us notice that the CFB mode can be rewritten into the form (4.5)-(4.6). The function $l_\theta$ is very simple since it merely reduces to a shift. The output function $h_\theta$ results from the composition of the block cipher (parametrized by its secret key $\theta$) and the filter function $h'$.

In the Maurer's approach, the SSSC is based on a cryptographically secure state-transition function $g_\theta$ as well as on a cryptographically secure output function $h_\theta$. Consequently, the resulting SSSC can be secure unless both functions are simultaneously unsecure. That differs from the CFB mode for which the security relies entirely on the security of the output function $h_\theta$ and so mostly on the block cipher function.

The second idea of the Maurer's principle consists in increasing the complexity by combining several finite automata in serial or in parallel or more generally by performing composition. As a result, many components that are relatively simple in terms of implementation complexity and memory size can be combined to form an SSSC realizing a very complicated function $\sigma_\theta^{ss}$ in the corresponding canonical representation (4.2). For a serial composition of multiple automata, the resulting memory size equals the sum of the memory size of each automaton. For a parallel composition of multiple automata, the resulting memory size equals the upper memory size. When implemented in hardware, parallelization leads to very high achievable encryption speed. An example of architecture involving four automata is depicted in FIG. 4.3.

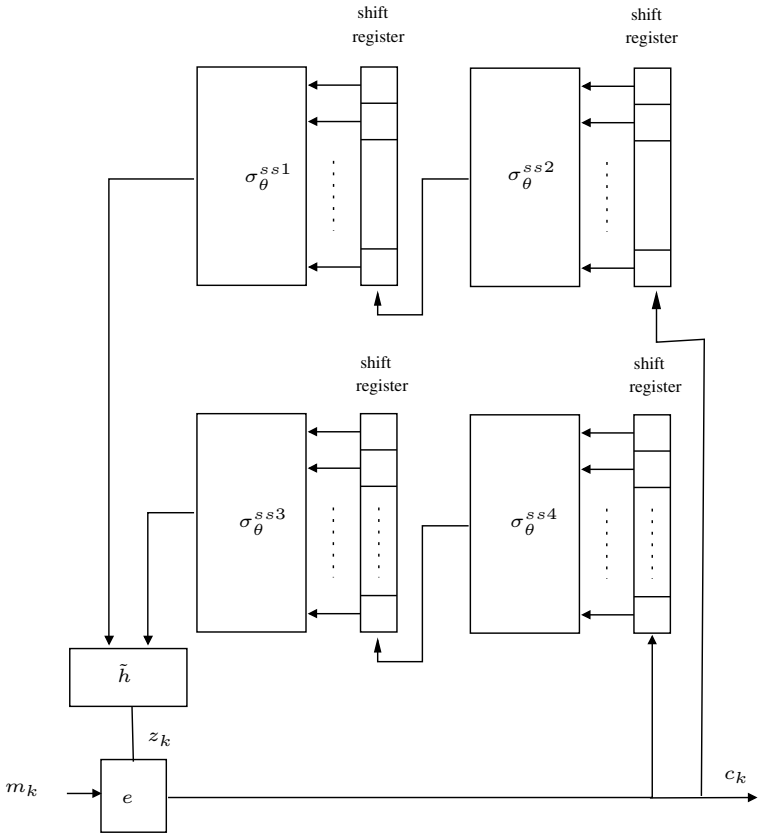


FIGURE 4.3 – Example of serial/parallel connection of four automata. The function $\tilde{h}$ combines the accessible automata outputs to deliver the keystream $z_k$

Two European projects have influenced the evolution of stream ciphers : the project NESSIE within the Information Society Technologies Programme of the European Commission which had started in 2000 and ended in 2004 followed by ECRYPT [5] launched on February 1st, 2004. Sponsored by ECRYPT, eSTREAM is a multi-year effort aiming at identifying promising both software and hardware oriented symmetric cryptosystems with proposals from industry to academia. Throughout the eSTREAM project, two fully specified algorithms have retained attention : SSS and Moustique. They are shortly described to illustrate how the general principle of Maurer is taken into account. As a matter of fact, only the first idea of Maurer consisting in resorting to an automaton with finite input memory has been adopted throughout these two examples. Indeed, as it turns out, the second idea is too general as is. These examples are also interesting in that they give us a better understanding in the way how the dynamical systems are "shaped" to guarantee the self-synchronization property.

#### 4.3.2.1 SSS

SSS is a software bit oriented cryptosystem which has been proposed in [HPRM04]. The corresponding block diagram is depicted on FIG. 4.4.

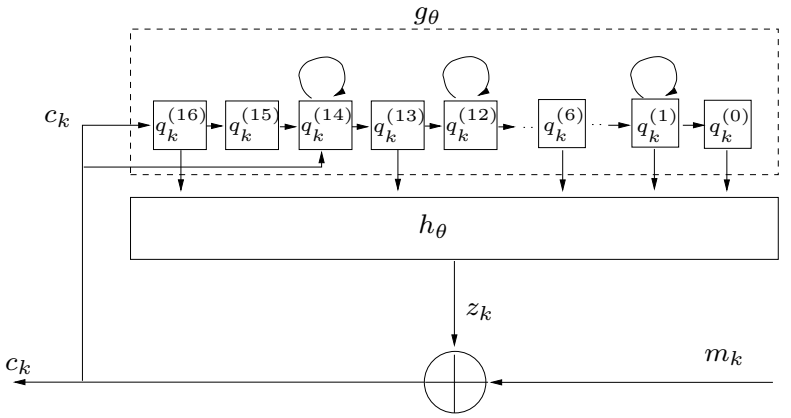


FIGURE 4.4 – Block diagram of SSS

The following notations are necessary to describe SSS.
– $x >>> n$ denoted the rotation of $n$ bits to the right of the word $x$
– $S_\theta(x) = SBOX_\theta(x_H) \oplus x$ with $x_H$ the most significant byte of the word $x$ is the XOR operation between $x$ and the result of $SBOX_\theta$ which is a combination of two S-boxes implementing nonlinear substitutions called Skipjack S-box and Q-box and parametrized by the secret key $\theta$

The keystream generator obeys (4.5). The dimension of the state vector $q_k$ equals $n = 17$ that is the number of shift registers. Each component $q_k^{(j)}$ assigned to a

5. website available at `http://www.ecrypt.eu.org/stream/`

shift register obeys an independent dynamics $g_\theta^j$ :

$$\begin{array}{rcl} q_{k+1}^{(16)} & = & c_k \\ q_{k+1}^{(j)} & = & q_k^{(j+1)} \ (j = 0, 2, \ldots, 11, 13, 15) \\ q_{k+1}^{(14)} & = & q_k^{(15)} + S_\theta(c_k >>> 8) \\ q_{k+1}^{(12)} & = & S_\theta(q_k^{(13)}) \\ q_{k+1}^{(1)} & = & q_k^{(2)} >>> 8 \end{array} \tag{4.7}$$

The initial state of the shift register number 16 fulfills $q_1^{(16)} = c_0$. Furthermore, insofar as the state $q_{k+1}^{(j)}$ $(j = 1, \ldots, 15)$, at time $k+1$, depends on the state $q_k^{(j+1)}$ at time $k$ (triangular feature), thus after 16 iterations, the internal state $q_k$ will depend exclusively on the 16 past ciphertexts $c_{k-i}$. Hence for all $k \geq 16$ there exists a function $l_\theta$ fulfilling

$$q_k = l_\theta(c_{k-16}, \ldots, c_{k-1}) \tag{4.8}$$

The output function $h_\theta$ delivering the keystream $z_k$ is defined as :

$$z_k = h_\theta(q_k) = A_\theta >>> 8 \oplus q_k^{(0)} \tag{4.9}$$

with $A_\theta = S_\theta\Big(S_\theta(q_k^{(0)} + q_k^{(16)}) + q_k^{(1)} + q_k^{(6)} + q_k^{(13)}\Big)$

Finally, combining the equations (4.8) and (4.9), the keystream generator can be equivalently rewritten in the SSSC canonical form (4.2) :

$$\begin{array}{rl} z_k & = h_\theta(l_\theta(c_{k-16}, .., c_{k-1})) \\ & = \sigma_\theta^{ss}(c_{k-16}, .., c_{k-1}) \end{array} \tag{4.10}$$

and guarantees the self-synchronization property.

The encryption function $e$ and decryption function $d$ follow the classical rules described in Subsection 4.1.2 where $\oplus$ is viewed in this case as a componentwise addition over the 2-element field.

#### 4.3.2.2 Moustique

Another interesting SSSC, called Moustique, which follows the first idea in the Maurer's approach, has been proposed in [DK05b]. It is a revisited version of two former algorithms called Mosquito and Knot. Unlike SSS, it is an hardware bit oriented algorithm. Furthermore, although the structure still relies on the automaton (4.5) which must have a finite memory, a different "shape" for the state transition function $g_\theta$ is provided to guarantee self-synchronizing property. Moreover, the output function is designed through the concept of *pipelining*. Those two facts are explicited below.

For Moustique, the dimension of the state vector $q_k$ in (4.5) equals $n = 96$.

As far as the state transition function $g_\theta$ is concerned, each component $q_k^{(j)}$ obeys a dynamics $g_\theta^j$ in the form :

$$q_{k+1}^{(j)} = g_\theta^j(q_k^{(j-1)}, q_k^{(j-2)}, ..., q_k^{(1)}, c_k) \; j = 1, \ldots, n \tag{4.11}$$

The $j^{th}$ component of $q_{k+1}$ does no longer depend exclusively on one component of $q_k$ (as it is for SSS), but it depends actually on several components of $q_k$, especially $q_k^{(l)}$ with $l < j$. The function $g_\theta$ has however, similarly to SSS, a triangular feature and ensures $q_k$ to be independent of the initial condition $q_0$ after $n$ iterations. Similarly to SSS, there exists thereby a function $l_\theta$ which expresses (4.11) in a different but strictly equivalent way for $k \geq n$ and depends exclusively on a finite number of past ciphertexts $c_{k-i}$

$$q_k = l_\theta(c_{k-n}, ..., c_{k-1}) \tag{4.12}$$

The output function is made up of a composition of $b_s = 9$ functions. Unlike SSS, the output function is pipelined (see FIG. 4.5). That means that the keystream is computed in a sequential way and the computation involves $b_s = 9$ successive stages. Each stage corresponds to a specific function $s_i$ $(i = 0, \ldots, b_s - 1)$ depending on the result of the previous stage. For the function $s_0$ one has $s_0(q_k) = q_k$. The keystream is computed from the state $q_k$ but is delivered at time $k + b_s$ :

$$z_{k+b_s} = s_8(s_7(...(s_0(q_k)))) = h(q_k) \tag{4.13}$$

Combining (4.12) and (4.13) gives $\sigma_\theta^{ss}$

$$\begin{aligned} z_{k+b_s} &= h(l_\theta(c_{k-n}, ..., c_{k-1})) \\ &= \sigma_\theta^{ss}(c_{k-n}, ..., c_{k-1}) \end{aligned} \tag{4.14}$$

As it turns out, the keystream generator can be again equivalently rewritten in the SSSC canonical form (4.2) and self-synchronization is guaranteed.

The pipeline is interesting in that it enables to increase the complexity of the output function while a single clock cycle is still needed to deliver the running key. Indeed the computation of each function $s_i$ is parallelized. That induces a delay $b_s$ between the plaintext and the corresponding ciphertext. Notice that none of the function $s_i$ depend on the secret key $\theta$. Actually, the output function $h_\theta$ in (4.5) should be rewritten as a non-parametrized function $h$.

Similarly to SSS, the encryption function $e$ and decryption function $d$ follow the classical rules described in Subsection 4.1.2.

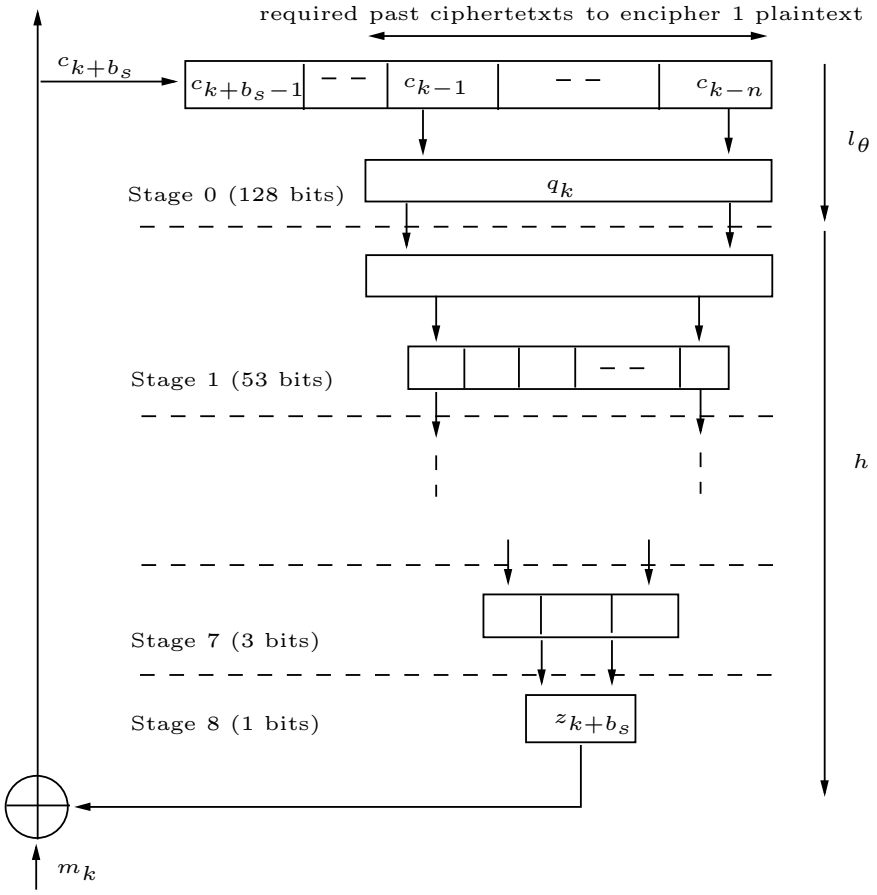


FIGURE 4.5 – Block diagram of Moustique. The functions $s_i$ deliver a quantity of decreasing size : from 128 bits for the stage 0 to a single bit for the last stage 8

## 4.4 Identification vs security of the message-embedding

### 4.4.1 General consideration

An essential issue for the validation of ciphers is the cryptanalysis that is the study of attacks against cryptographic schemes in order to reveal their possible weakness. A fundamental assumption in cryptography first stated by A. Kerkhoff in ([DK02]), is that any unauthorized person (called adversary or eavesdropper) knows all the details of the cipher, including the algorithm and its implementation, except the secret key. As a result, as far as the parameters of (3.1) are expected to act as the secret key, the security is directly related to the complexity of retrieving the parameters $\theta$.

It is usual to assume that the eavesdropper has the opportunity of controlling the input of the cipher, namely the plaintext, and analyzing the corresponding ciphertext (the attack is called chosen plaintext attack). In our context, if the dynamical system (3.1) is considered as a cipher, that means that the pair $(m_k, y_k)$ is assumed to be known by the eavesdropper. The recovery of $\theta$ can only be achieved through an identification procedure based the input/output model of (3.1). The identification procedure detailed in Chapter 3 is thereby nothing but a so-called algebraic attack.

Besides, it worth emphasizing that a cipher must face at least the most basic attack, i.e. the brute force attack. This attack consists in trying exhaustively every possible parameter value in the parameter space of the secret key (which is in practice a finite space). The quicker the brute force attack, the weaker the cipher. Consequently, the worst situation for the eavesdropper and the best for the security arises when, for known plaintexts and corresponding ciphertext sequences, only one solution in the parameters of the cipher exists. The unicity is directly related to the notion of parametric identifiability.

As a result, we conclude that the most relevant parameters of a system to act as the secret key are the ones which are identifiable. Such a result might appear as paradoxical at first glance because of a possible misunderstanding on the meaning of "identifiable". Actually, identifiability means unicity in the parameters. Such a paradox has been highlighted in [AMB06].

### 4.4.2 Particularization for switched linear systems

We recall the Equation (3.1) in Chapter 3 of the message-embedded cryptosystem particularized for switched linear systems :

$$\left\{ \begin{array}{lcl} x_{k+1} & = & A_{\sigma(k)}x_k + B_{\sigma(k)}m_k \\ y_k & = & C_{\sigma(k)}x_k + D_{\sigma(k)}m_k \end{array} \right.$$

When particularized for switched linear systems, the aforementioned consideration on unicity yields the following proposition :

**Proposition 9** *The secret key $\theta$ must be the set of entries of $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ and $(D_j)_{1\leq j\leq J}$ of (3.1) which can be deduced from $c(\sigma_t)$ and the $a_j(\sigma_t)$'s in a unique way.*

Actually, the security is related to the complexity of the underlying identification procedure (see Chapter 3). Clearly the identification procedure is much more complex when $\sigma_t$ is not accessible. Thus the secret key $\theta$ must be determined so that the eavesdropper has no other choice than resorting to the second identification procedure. As a result, $\sigma_t$ must not be directly accessible and the following proposition must be thereby fulfilled :

**Proposition 10** *The switching rule $\sigma$ must depend on $\theta$.*

We can assess the security in terms of the complexity of the required algebraic computations to identify $\theta$. The most important task in the identification procedure related to the case when the switching sequences $\sigma_t$ are not accessible is the computation of the coefficients $h_N$ through (3.25). In practice, the kernel (null space) is obtained through a Gaussian-Bareiss elimination of which complexity is $O(min(N'M_N^2, N'^2M_N))$. The lower bound of $N'$ being $M_N - 1$, when $M_N$ is large enough, the complexity can be approximated by $O(M_N^3)$. The expansion rate of $M_N$ and complexity for the difference values of $N$ and $K$ are depicted respectively in FIG. 4.6 and FIG. 4.7.

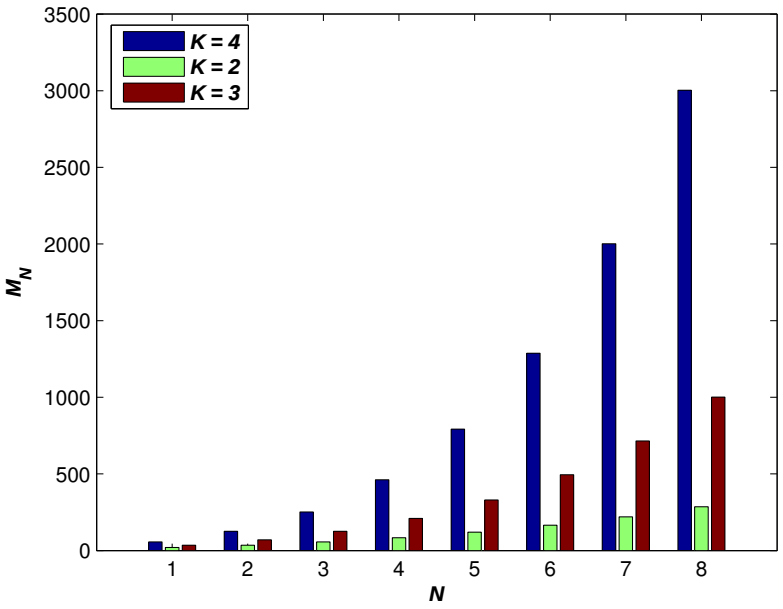


FIGURE 4.6 – $M_N$ versus $N$ for difference values of $K$

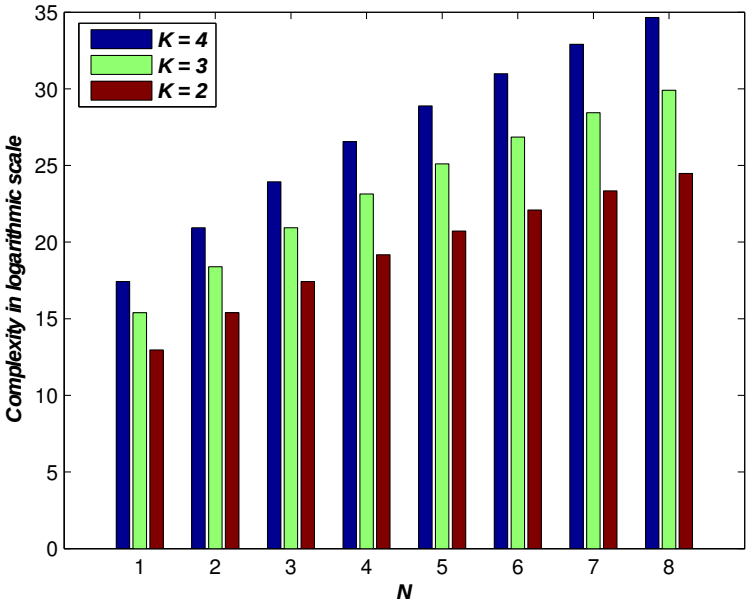


FIGURE 4.7 – The evolution of complexity when $N$ varies for difference values of $K$

## 4.5 Conclusion

The main results of this Chapter are summed up. The message-embedding scheme may act as a self-synchronizing stream cipher (SSSC) from a structural point of view under flatness condition. In standard SSSC ciphers, in order to guarantee a self-synchronizing property, the state transition function must have a particular feature : triangular. On the other hand, flatness confers a self-synchronizing property without such a constraint and appears as an alternative for the design of SSSC. Identifiability is the necessary required property such that the parameters of the message-embedding may be involved in the secret key. Identification consists of an algebraic attack in the context of secure communication. For switched linear systems, the complexity to identify the parameters can increase significantly with the number of modes.

# Conclusion

Various cryptosystems, corresponding to different ways of hiding a message, have drawn the attention of the researchers since the 90's. The message-embedding appears to be the most attractive as the synchronization between the transmitter and the receiver can be guaranteed without any restriction on the rate of variation of the message to be encrypted and a single channel is required.

However, if a digital application is sought (hardware implementation in e.g. FPGA or DSP), resorting to a map which directly takes value in a finite set whose range is identical than the one of the data is a better solution. As a result, the message-embedding over a finite field must be considered. It has been stressed that the only acceptable systems for which left inversion at the receiver side can be carried out are flat ones. Methodologies for the design of a left inverse system over finite fields have been provided. Besides, it has been highlighted that flat systems are structurally equivalent to conventional stream ciphers called Self Synchronizing Stream Ciphers and the use of flat systems appears as a new solution for the design of Self Synchronizing Stream Ciphers. A particularization for switched linear systems has been made. This special class of hybrid systems obeys the Shamir's suggestions [KS04] consisting of mixing different algebra.

If a practical and viable application of such dynamical systems is sought, the security aspect must be taken into account. Chaos-based cryptographic primitives were most often considered as secure exclusively because of the complexity of the dynamics which is exhibited. Over a finite field, chaos does no longer make sense. The security has been assessed here in terms of the parameters recovering task complexity. Thus, a special identification procedure over finite fields and its related complexity has been provided.

Finally, the message-embedding scheme is currently being implemented in practice. The testbench, based on the FPGA, consists of a four-dimensional switched linear system which have four modes and works on the binary finite field. Preliminary results shows that the system is operating well. The overall tasks for encrypting/decrypting the video signal in the real time are fulfilled.

Let us now address possible perspectives.

An essential issue for the validation of cryptosystems is the cryptanalysis, that is the study of attacks against cryptographic schemes in order to reveal their possible weaknesses. The consideration in the design of the possible attacks and their complexity dictates the way how the secret key must be defined. We quote some of cryptanalytic approaches which deserve attention in the context of the message-embedding.

The core of an SSSC is the ciphering function. Its complexity can be assessed through the "distance" from a given function having low algebraic degree (see [GM05] for the details). If the "distance" is not large enough, then there exists decoding algorithms that are able to reconstruct the whole low degree approximation of the ciphering function and provide thereby an estimation of the plaintext.

Furthermore, it can be proved that a sufficient condition for an SSSC to be secure is that the adversary cannot distinguish the ciphering function from a random one. Indeed, in this case, the cryptanalyst has no information at all on the keystream. The existence of a distinguisher is a weakness in the ciphering function.

The question whether switched linear systems could be good candidates for designing cryptosystems deserves deeper insights. Piecewise nonlinearities are likely to be not resistant enough and others nonlinearities should be considered while keeping the hybrid aspect.

If the secret key is embedded in a device such as a smart card or an electronic component, an adversary who has temporarily access to the device may try to recover the secret key through physical measures such as time, power consumption, glitch and so on. The consideration of these attacks, known as side-channels attacks, is a modern topic of great interest at the moment. As a result, the issue of implementation which could resist such attacks must be seriously addressed and can constitute interesting further works.

# Appendix

# Appendix A

# Lyapunov exponents

Consider an autonomous dynamical system :

$$x_{k+1} = f(x_k) \tag{A.1}$$

where $x_k \in \mathbb{R}^n$. We assume that the trajectory emanating from an initial condition $x_0$ has reached an attractor ($x_k$ is bounded).

**Case** $n = 1$
Let $x_0$, $x'_0$ denote two nearby initial conditions. If two trajectories with iterates $x_k$ and $x'_k$ evolve exponentially after $k$ iterations, we have :

$$|x'_k - x_k| = |x'_0 - x_0|e^{k\lambda}$$

$\lambda$ corresponds to the divergence rate of two trajectories and is given as :

$$\lambda = \frac{1}{k} \ln \left| \frac{x'_k - x_k}{x'_0 - x_0} \right|$$

If $x_0$ and $x'_0$ are very close, their difference $\epsilon = |x'_0 - x_0|$ tends toward 0, we define :

$$\lambda_L = \lim_{k\to\infty} \frac{1}{k} \lim_{\epsilon\to 0} \ln \left| \frac{x'_k - x_k}{x'_0 - x_0} \right|$$

This yields :

$$\lambda_L = \lim_{k\to\infty} \frac{1}{k} \lim_{\epsilon\to 0} \ln \left| \frac{x'_k - x_k}{x'_{k-1} - x_{k-1}} \frac{x'_{k-1} - x_{k-1}}{x'_{k-2} - x_{k-2}} \cdots \frac{x'_1 - x_1}{x'_0 - x_0} \right|$$

and

$$\lambda_L = \lim_{k\to\infty} \frac{1}{k} \lim_{\epsilon\to 0} \sum_{i=0}^{k-1} \ln \left| \frac{x'_{i+1} - x_{i+1}}{x'_i - x_i} \right| = \lim_{k\to\infty} \frac{1}{k} \lim_{\epsilon\to 0} \sum_{i=0}^{k-1} \ln \left| \frac{f(x'_i) - f(x_i)}{x'_i - x_i} \right|$$

Finally, we have :

$$\lambda_L = \lim_{k\to\infty} \frac{1}{k} \sum_{i=0}^{k-1} \ln \left| \frac{df(x_i)}{d(x_i)} \right| \tag{A.2}$$

The quantity $\lambda_L$ is called Lyapunov exponent. $\lambda_L$ measures the convergence/divergence rate of two distinct trajectories starting from two nearby initial conditions. If $\lambda_L$ is positive, two trajectories are divergent and the dynamical system is chaotic. In particular, it is sensitive to initial conditions.

**Case** $n$>1

There are $n$ Lyapunov exponents $\lambda_L^{(j)}$ $(j = 1, \ldots, n)$. Each one characterizes the convergence/divergence rate of two distinct trajectories starting from two nearby initial conditions along $n$ orthogonal directions.

For computing the Lyapunov exponent, we start from an initial point $x_0 \in \mathbb{R}^n$ and characterize the infinitesimal behavior near the point $x_k$ through the first derivative matrix

$$Df(x_i) = \begin{bmatrix} \frac{\partial f_1(x_i)}{\partial x_i^{(1)}} & \cdots & \frac{\partial f_1(x_i)}{\partial x_i^{(n)}} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n(x_i)}{\partial x_i^{(1)}} & \cdots & \frac{\partial f_n(x_i)}{\partial x_i^{(n)}} \end{bmatrix}$$

Denote $J_k = Df(x_{k-1}) \cdots Df(x_0)$ with $J_0 = Df(x_0)$. The Lyapunov exponent is computed as :

$$\lambda_L = \lim_{k \to \infty} \frac{1}{k} \ln \left| eig \left( J_k J_k^T \right) \right| \tag{A.3}$$

The square roots of $n$ eigenvalues of the matrix $J_k J_k^T$ stand for the length of $n$ axes of an image ellipsoid. They qualify the amount of shrinking and stretching due to the dynamic near the orbit beginning at $x_0$. When $k$ is large, the computation of the Lyapunov exponent in (A.3) is delicate because $J_k J_k^T$ is often a bad conditioned matrix (very small and large eigenvalues). To tackle this problem, we need to compute $J_k$ recursively and perform a normalization. It is now detailed.

We are given an initial orthonormal basis $\{w_1^{(0)}, \cdots, w_n^{(0)}\}$ in $\mathbb{R}^n$ and $J_0$.

Each step $i$ $(i = 1, \cdots, k)$ involves the following operations :

– compute the quantity

$$J_i = Df(x_{i-1}) \cdots Df(x_0)$$

and the vector

$$z_j^{(i)} = J_i w_j^{(i-1)} \quad j = (1, \cdots, n)$$

– derive $\{w_j^{(i)}\}_{j=1,\ldots,n}$, the orthogonal set of $\{z_j^{(i)}\}_{j=1,\ldots,n}$ obtained from the Gram-Schmidt orthogonalization algorithm. As a result, $\{w_j^{(i)}\}_{j=1,..,n}$ measures the one step growth in the direction $j$. The total expansion rate in the direction $j$ after $k$ steps called Lyapunov numbers is defined as $e_j^k = ||w_j^{(k)}|| \ldots ||w_j^{(1)}||$.

The Lyapunov Exponent $\lambda_L^{(j)}$ is given by :

$$\lambda_L^{(j)} = ln(e_j^k)^{1/k} = \frac{1}{k} \sum_{i=1}^{k} \ln(||w_j^{(i)}||)$$

For the system (A.1), the attractor is chaotic if there exists at least one $\lambda_L^{(j)} > 0$

# Appendix B

# Algebra

The reader can refer to [Lan02],[Cal98] for useful algebra material.

**Law of composition** Let $\mathbb{S}$ be a set. A mapping $\mathbb{S} \times \mathbb{S} \mapsto \mathbb{S}$, from $\mathbb{S}$ into itself, is called *law of composition*. Let $x, y$ be two elements in $\mathbb{S}$. We often have 2 laws of composition : addition $x + y$, multiplication $x.y$
If $(x + y) + z = x + (y + z)$ $\forall x, y, z \in \mathbb{S}$, we say that the addition is *associative*.
If $(x.y).z = x.(y.z)$ $\forall x, y, z \in \mathbb{S}$, we say that the multiplication is *associative*.
If $x + y = y + x$ $\forall x, y \in \mathbb{S}$, we say that the addition is *commutative*.
If $x.y = y.x$ $\forall x, y \in \mathbb{S}$, we say that the multiplication is *commutative*.

**Remark 16** *Since the image of the law of composition is also in* $\mathbb{S}$*, the law of composition implies the "closure" property.*

**Unit element of a law of composition** :
An element $e$ of $\mathbb{S}$, such that $x.e = x = e.x$ $\forall x \in \mathbb{S}$, is called *unit element* of multiplication law.
An element $e$ of $\mathbb{S}$, such that $x + e = x = e + x$ $\forall x \in \mathbb{S}$, is called *unit element or zero element* of addition law.

**Monoid** A monoid is a set with a law of composition which is associative, and having a unit element.

**Remark 17** *When the law of composition is commutative, we have a commutative monoid or abelian monoid.*

**Example 1** $(\mathbb{N}, +)$ *is an abelian monoid .* $(\mathbb{N}^+, +)$ *is not a monoid since unit element does not exist.*

**Group** A group $\mathbb{G}$ is a *monoid*, such that for every element $x \in \mathbb{G}$ there exists an element $y \in \mathbb{G}$ such that $xy = yx = e$. Such an element $y$ is called an inverse for $x$. In addition, this inverse is unique.

**Remark 18** *If* $\mathbb{G}$ *is an abelian/commutative monoid with unique inverse element,* $\mathbb{G}$ *is called an abelian/commutative group.*

**Example 2** $(\mathbb{Z}, +)$ *is an abelian group but* $(\mathbb{Z}, \times)$ *is not since inverse element does not exist.*

**Cyclic groups** A group $\mathbb{G}$ is defined to be *cyclic* if there exists an element $a \in \mathbb{G}$ such that every element of $\mathbb{G}$ (written multiplicatively) is of the form $a^n$ for some integer $n$. If $\mathbb{G}$ is written additively, then every element of a cyclic group is of the form $na$. One calls $a$ a cyclic generator.

**Example 3** $(\mathbb{Z}, +)$ *is an additive cyclic group with generator* 1*, and also with generator* $-1$*. There are no other generators.* $(\mathbb{Z}_p, \times)$ *is a group but it is not a cyclic group since we cannot find a generator a so that we can write every element in the form* $a^n$*.*

**Subgroup** Let $\mathbb{G}$ be a group. A subgroup $\mathbb{H}$ of $\mathbb{G}$ is a subset of $\mathbb{G}$ containing the unit element, and such that $\mathbb{H}$ is closed under the law of composition and inverse (i.e. if $x \in \mathbb{H}$ then $x^{-1} \in \mathbb{H}$). A subgroup is called trivial if it consists of the unit element alone. The intersection of an arbitrary non-empty family of subgroups is a subgroup.

**Ring** A ring $\mathbb{A}$ is a set, together with *two laws of composition* called multiplication and addition respectively, and written as a product and as a sum respectively, satisfying the following conditions :
– With respect to addition, $\mathbb{A}$ is a commutative group.
– The multiplication is associative, and has a unit element.
– For all $x, y, z \in \mathbb{A}$ we have $(x + y)z = xz + yz$

**Remark 19** *If the multiplication law is commutative (ie. both associative and commutative), we have a abelian/commutative ring.*

**Example 4** $\mathbb{Z}_p$ *is an abelian ring because it is an abelian group with the addition and the multiplication law.*

Let $\mathbb{A}$ be a ring, and let $\mathbb{U}$ be the set of elements of $\mathbb{A}$ which have both a right and left inverse. Then $\mathbb{U}$ is a *multiplicative group* (each element of $\mathbb{U}$ have multiplicative inverse). Indeed, if $a$ has a right inverse $b$, so that $ab = 1$, and a left inverse $c$, so that $ca = 1$, then $cab = b$, whence $c = b$, and we see that $c$ (or $b$) is a two-sided inverse, and that $c$ itself has a two-sided inverse, namely $a$. Therefore $\mathbb{U}$ satisfies all the axioms of a multiplicative group, and is called the group of units of $\mathbb{A}$. It is sometimes denoted by $\mathbb{A}*$, and is also called the group of invertible elements of $\mathbb{A}$.

**Division ring** A ring $\mathbb{A}$ such that $1 \neq 0$, and such that every non-zero element is invertible is called a division ring.

**Example 5** *$\mathbb{Z}_p$ where $p$ is a prime number, is a division ring but $\mathbb{Z}_n$, with $n$ a non zero natural number, is not. Let's consider an abelian ring $\mathbb{Z}_k$ and $a$ is an arbitrary element in $\mathbb{Z}_k$. Let $b$ be an inverse element of $a$. Then $ba = 1 \ mod \ k$. This yields $ba + kc = 1$ and implies that $b$ exists if and only if $gcd(a, k) = 1 \ \forall a \in \mathbb{Z}_k$. So we have the solution.*

**Field** A commutative division ring is called a *field*. We observe that by definition, a field contains at least two elements, namely 0 and 1.

**Ideal** A left ideal $I$ in a ring $\mathbb{A}$ is a subset of $\mathbb{A}$ which is a subgroup of the additive group of $\mathbb{A}$ (ie. has the same composition law), such that $\mathbb{A} \times I \subset I$. We have the same definition for right ideal. On the commutative/abelian ring, every left or right ideal is a two-sided ideal. The two-sided ideal is called simply ideal. Note that (0) and $\mathbb{A}$ itself are ideal.

If $\mathbb{A}$ is a ring and $a \in \mathbb{A}$, then $Aa = I$ is a left ideal, called principal. We say that $a$ is a generator of $I$ (over $\mathbb{A}$). More generally, let $a_1, ..., a_n$ be elements of $\mathbb{A}$. We denote by $(a_1, ..., a_n)$ the set of elements of $\mathbb{A}$ which can be written in the form $x_1a_1 + ... + x_na_n$ $(x_i \in \mathbb{A})$. $(a_1, ..., a_n)$ is a left ideal and $a_1, ..., a_n$ are generators of left ideal.

**Example 6** *Consider the abelian ring $\mathbb{Z}$. Each number in $\mathbb{Z}$ is an ideal. Let us choose $n = 2 \in \mathbb{Z}$. Since $\mathbb{Z}$ is abelian, the set of even number $\{2\mathbb{Z}\}$ is an ideal, called principal (because the set $\{2\mathbb{Z}\}$ is generated by a unique generator). The number 2 is called generator of $I = 2\mathbb{Z}$ over $\mathbb{Z}$.*

# Appendix C

# Gaussian elimination

## C.1 Gauss-Bareiss elimination

We recall the Sylvester's identity Theorem and its application in the Gaussian-Bareiss elimination method described in [Bar68].

**Sylvester identity**
Consider a matrix $A = (a_{ij}) \in \mathbb{A}^{n\times n}$ where $\mathbb{A}$ is an arbitrary abelian ring. For $k < i, j \leq n$, the $(k+1)$ order minor is the determinant of the matrix constructed by the first $k$ rows and first $k$ columns and augmented with the $i$-th row and $j$-th column of $A$.

$$a_{i,j}^{[k]} = \begin{vmatrix} a_{11} & \dots & a_{1k} & a_{1j} \\ \vdots & & \vdots & \vdots \\ a_{k1} & \dots & a_{kk} & a_{kj} \\ a_{i1} & \dots & a_{ik} & a_{ij} \end{vmatrix}$$

It can be noted that the $(k+1)$ order minor $a_{i,j}^{[k]}$ is the determinant of a $(k+1)\times(k+1)$ matrix.

**Example 7** *: consider a* $4 \times 4$ *matrix*

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

*All the* 3 *order minors are given by :*

$$a_{3,3}^{[2]} = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} \quad a_{3,4}^{[2]} = \begin{vmatrix} a_{11} & a_{12} & a_{14} \\ a_{21} & a_{22} & a_{24} \\ a_{31} & a_{32} & a_{34} \end{vmatrix}$$

$$a_{4,3}^{[2]} = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{41} & a_{42} & a_{43} \end{vmatrix} \quad a_{4,4}^{[2]} = \begin{vmatrix} a_{11} & a_{12} & a_{14} \\ a_{21} & a_{22} & a_{24} \\ a_{41} & a_{42} & a_{44} \end{vmatrix}$$

**Theorem 6** *(Sylvester's identity) Given a matrix $A \in \mathbb{A}^{n\times n}$ where $\mathbb{A}$ is an arbitrary abelian ring. We suppose that the $k$ order minor of $A$, $a_{k,k}^{[k-1]}$, is different to zero $\forall k \geq 1$. We have :*

$$|A|(a_{k,k}^{[k-1]})^{n-k-1} = \begin{vmatrix} a_{k+1,k+1}^{[k]} & a_{k+1,k+2}^{[k]} & \cdots & a_{k+1,n}^{[k]} \\ a_{k+2,k+1}^{[k]} & \vdots & & \vdots \\ \vdots & & & \\ a_{n,k+1}^{[k]} & a_{n,k+2}^{[k]} & & a_{n,n}^{[k]} \end{vmatrix}$$

*with $a_{i,j}^{[0]} = a_{i,j}$*

**Proof 6** *Matrix $A$ is first divided into 4 block sub-matrices*

$$A = \begin{vmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{vmatrix}$$

*with $A_{1,1} \in \mathbb{A}^{k\times k}$, $A_{1,2} \in \mathbb{A}^{k\times(n-k)}$, $A_{2,1} \in \mathbb{A}^{(n-k)\times k}$, $A_{2,2} \in \mathbb{A}^{(n-k)\times(n-k)}$.*

*We have the following relation*

$$A = \begin{pmatrix} A_{1,1} & \mathbf{0} \\ A_{2,1} & \mathbf{1}_{n-k} \end{pmatrix} \begin{pmatrix} \mathbf{1}_k & A_{1,1}^{-1} \cdot A_{1,2} \\ \mathbf{0} & A_{2,2} - A_{2,1} \cdot A_{1,1}^{-1} \cdot A_{1,2} \end{pmatrix}$$

*The determinant of $A$ verifies :*

$$|A| = |A_{1,1}| \cdot |A_{2,2} - A_{2,1} \cdot A_{1,1}^{-1} A_{1,2}| \tag{C.1}$$

*and*

$$|A||A_{1,1}|^{n-k-1} = |A_{1,1}|^{n-k} \cdot |A_{2,2} - A_{2,1} \cdot A_{1,1}^{-1} A_{1,2}|$$

*Notes that $|c \cdot M| = c^n |M|$ with $c$ is an arbitrary constant and $M \in \mathbb{A}^{n\times n}$. Hence, we have :*

$$|A||A_{1,1}|^{n-k-1} = ||A_{1,1}|(A_{2,2} - A_{2,1} \cdot A_{1,1}^{-1} A_{1,2})| \tag{C.2}$$

*where $(A_{2,2} - A_{2,1} \cdot A_{1,1}^{-1} A_{1,2}) \in \mathbb{A}^{n-k}$.*

*Consider now the $k$ order minor, the relation* (C.1) *still holds in this case*

$$a_{i,j}^{[k]} = \begin{vmatrix} a_{11} & \dots & a_{1k} & a_{1j} \\ \vdots & & \vdots & \vdots \\ a_{k1} & \dots & a_{kk} & a_{kj} \\ a_{i1} & \dots & a_{ik} & a_{ij} \end{vmatrix} = |A_{1,1}| \cdot |a_{ij} - r_i^T \cdot A_{1,1}^{-1} \cdot c_j| \tag{C.3}$$

*where* $r_i = (a_{i1}, \dots, a_{ik}) \in \mathbb{A}^{1\times k}$, $c_j^T = (a_{1j}, \dots, a_{kj}) \in \mathbb{A}^{1\times k}$
*Since* $(a_{ij} - r_i^T \cdot A_{1,1}^{-1} \cdot c_j)$ *is a scalar, we have :*

$$a_{i,j}^{[k]} = \begin{vmatrix} a_{11} & \dots & a_{1k} & a_{1j} \\ \vdots & & \vdots & \vdots \\ a_{k1} & \dots & a_{kk} & a_{kj} \\ a_{i1} & \dots & a_{ik} & a_{ij} \end{vmatrix} = |A_{1,1}| \cdot (a_{ij} - r_i^T \cdot A_{1,1}^{-1} \cdot c_j) \tag{C.4}$$

*For all* $i, j \in [k+1, n]$, *we have the following relation :*

$$\left(a_{i,j}^{[k]}\right) = |A_{1,1}|(A_{2,2} - A_{2,1} \cdot A_{1,1}^{-1} A_{1,2})$$

*Then*

$$\left|(a_{i,j}^{[k]})\right| = ||A_{1,1}|(A_{2,2} - A_{2,1} \cdot A_{1,1}^{-1} A_{1,2})|$$

*From* (C.2) *and* $|A_{1,1}| = a_{k,k}^{[k-1]}$ *the proof of Theorem 6 is completed.*

**Example 8** *: Consider a* $4 \times 4$ *matrix*

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

*Apply Theorem 6 with* $k = 3$, *that is considering the* 3 *order minor,* $a_{3,3}^{[2]}$, *we have :*

$$|A|(a_{3,3}^{[2]})^0 = \left| \; a_{4,4}^{[3]} \; \right| \tag{C.5}$$

*Apply Theorem 6 with* $k = 4$, *that is considering the* 4 *order minor,* $a_{4,4}^{[3]}$, *we have :*

$$|A| = a_{4,4}^{[3]} = \begin{vmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{vmatrix}$$

*In others words, the determinant of a* $n \times n$ *matrix is nothing but its* $n$ *order minor which is unique.*

*Apply Theorem 6 with $k = 2$, that is considering the 2 order minor $a^{[1]}_{2,2}$, we have :*

$$|A|a^{[1]}_{2,2} = \begin{vmatrix} a^{[2]}_{3,3} & a^{[2]}_{3,4} \\ a^{[2]}_{4,3} & a^{[2]}_{4,4} \end{vmatrix} \tag{C.6}$$

It is worthy to note that the determinant of matrix $A$ in Theorem 6 corresponds to, $a^{[n-1]}_{n,n}$, the $n$ order minor. Theorem 6 still holds while substituting $|A|$ by $a^{[k+1]}_{i,j}$, the $(k+2)$ order minor. Consequently, we substitute $n$ by $k+2$. This yields :

$$a^{[k+1]}_{i,j} \cdot (a^{[k-1]}_{k,k})^{\left((k+2)-k-1\right)} = a^{[k+1]}_{i,j} \cdot a^{[k-1]}_{k,k} = \begin{vmatrix} a^{[k]}_{k+1,k+1} & a^{[k]}_{k+1,j} \\ a^{[k]}_{i,k+1} & a^{[k]}_{i,j} \end{vmatrix} \tag{C.7}$$

for $k+1 < i \leq n\ ,\ k+1 < j \leq n\ ,\ 0 \leq k < n-1$

Let us consider a sub-matrix $L = (l_{i,j}) \in \mathbb{A}^{m \times n}$ with $m < n$, constructed by the first $m$ columns of matrix $A$ where $l_{i,j}$ denotes the component at the $i$-th row and $j$-th column of matrix $L$. We have :

$$l_{i,j} = a_{i,j} \quad \text{for} \quad 1 \leq i \leq n\ ,\ 1 \leq j \leq m \tag{C.8}$$

Substituting $a_{i,j}$ in (C.7) by $l_{i,j}$ in (C.8), we obtain :

$$l^{[k+1]}_{i,j} \cdot l^{[k-1]}_{k,k} = \begin{vmatrix} l^{[k]}_{k+1,k+1} & k^{[k]}_{k+1,j} \\ l^{[k]}_{i,k+1} & l^{[k]}_{i,j} \end{vmatrix} \tag{C.9}$$

for $k+1 < i \leq n\ ,\ k+1 < j \leq m\ ,\ 0 \leq k < m-1$

The relation C.9 is very useful to construct the fraction free Gaussian-Bareiss which is described in the sequel

**Gaussian-Bareiss elimination method**

Gaussian-Bareiss elimination technique is an integer-preserving gaussian elimination. It is often used to solve linear equations. The algorithm works in a recursive way.

Consider an arbitrary matrix $L = (l_{ij}) \in \mathbb{A}^{n \times m}$ with $n \geq m$. Let us denote $L^{(0)} = (l_{ij})$. The algorithm computes recursively matrices which are denoted $L^{(k)}$ at each iteration $k$ and $l^{(k)}_{i,j}$ are the corresponding entries.

Assuming that the so-called pivot element $l^{[k-1]}_{k,k} \neq 0$, the matrix $L^{(k+1)}$ is constructed from $L^{(k)}$ according to :

$$\begin{cases} l^{(k+1)}_{i,j} = l^{(k)}_{i,j} & \text{if} \quad 1 \leq i \leq k+1\ ,\ 1 \leq j \leq m \\ \\ l^{(k+1)}_{i,j} = l^{[k+1]}_{i,j} = \dfrac{\begin{vmatrix} l^{[k]}_{k+1,k+1} & l^{[k]}_{k+1,j} \\ l^{[k]}_{i,k+1} & l^{[k]}_{i,j} \end{vmatrix}}{l^{[k-1]}_{k,k}} & \text{if} \quad k+1 < i \leq n\ ,\ 1 \leq j \leq m \end{cases} \tag{C.10}$$

The algorithm performs $m-1$ steps.

**Remark 20** *The update in the Eq. (C.10) is derived from the Eq. (C.9). Let us note that Eq. (C.9) is derived from the Sylvester's identity Theorem which actually applies for square matrices. Nevertheless, we can construct an augmented square matrix from L with dummy columns. Considering only the first m columns of this augmented matrix yields exactly Eq. (C.10).*

**Remark 21** *At step k, if the pivot element $l_{k+2,k+2}^{[k+1]} = 0$, it is necessary to switch the $(k+2)$-th row of the matrix $L^{(k+1)}$ with an arbitrary t-th row of $L^{(k+1)}$, $t \in \{k+3, \cdots, n\}$, for which $l_{t,k+1}^{[k+1]} \neq 0$ to prevent from a division by zero. The algorithm stops if we can not find any $l_{t,k+2}^{[k+1]} \neq 0$.*

**Remark 22** *Assume that the matrix L has two rows linearly dependent, say, the $(k+1)$-th row and the $(k+d)$-th row, then we have :*

$$l_{k+1,j}^{(0)} = s.l_{k+d,j}^{(0)}$$

*By induction, we can show that :*

$$l_{k+1,j}^{[k]} = s.l_{k+d,j}^{[k]}$$

*At step k, the value of the $(k+d)$-th rows of $A^{(k+1)}$*

$$l_{k+d,j}^{[k+1]} = \frac{\begin{vmatrix} l_{k+1,k+1}^{[k]} & l_{k+1,j}^{[k]} \\ l_{k+d,k+1}^{[k]} & l_{k+d,j}^{[k]} \end{vmatrix}}{l_{k,k}^{[k-1]}} = 0$$

**Remark 23** *Bareiss' elimination method works on a matrix whose entries belong to an arbitrary abelian ring $\mathbb{A}$. Let us observe that it still works work on a field $\mathbb{F}_p$.*

**Remark 24** *If $n < m$, the algorithm still holds.*

The Gaussian-Bareiss elimination method is summed up in the FIG. C.1 and FIG. C.2

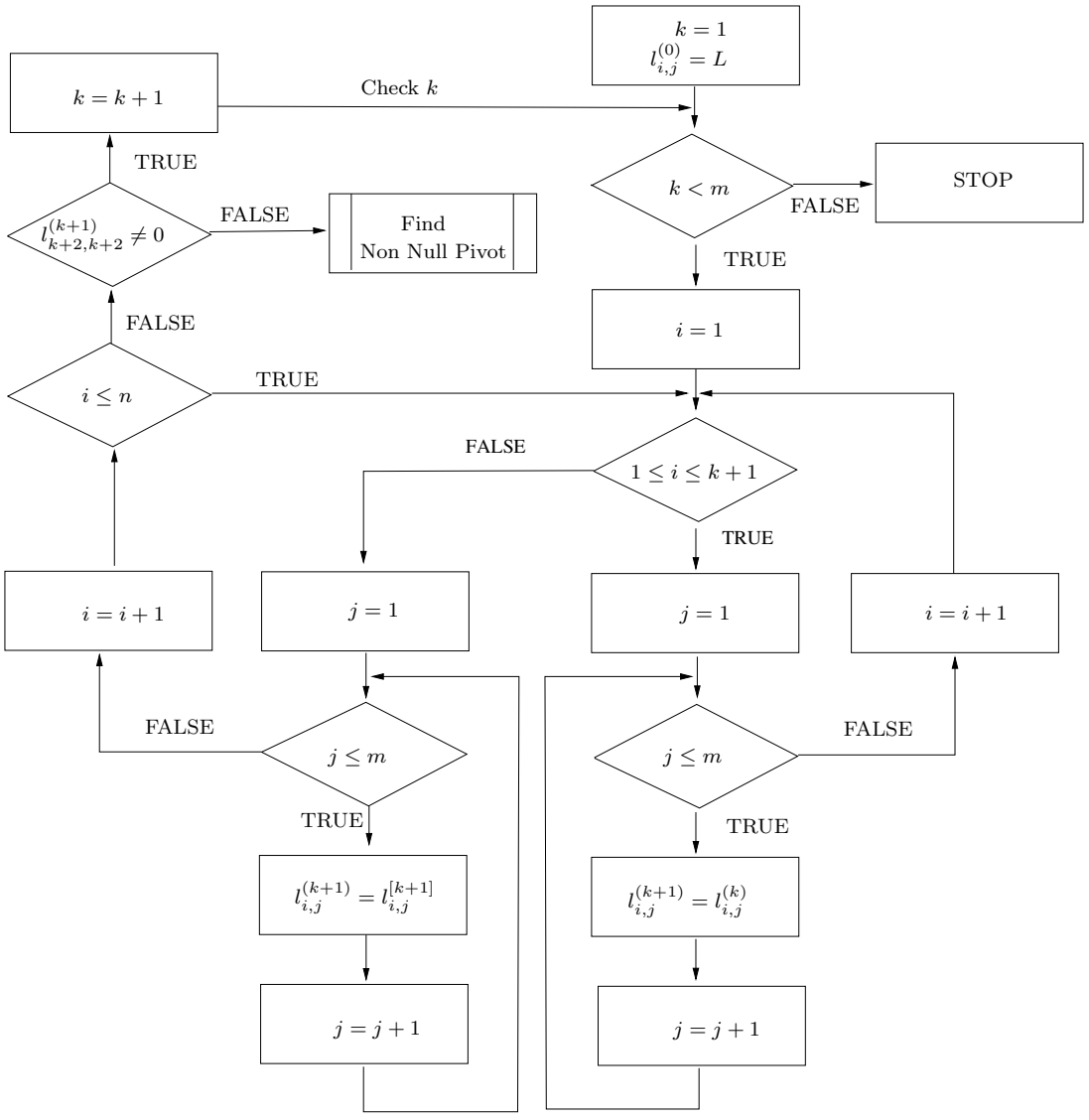


FIGURE C.1 – Diagram detailing the Gaussian-Bareiss elimination method

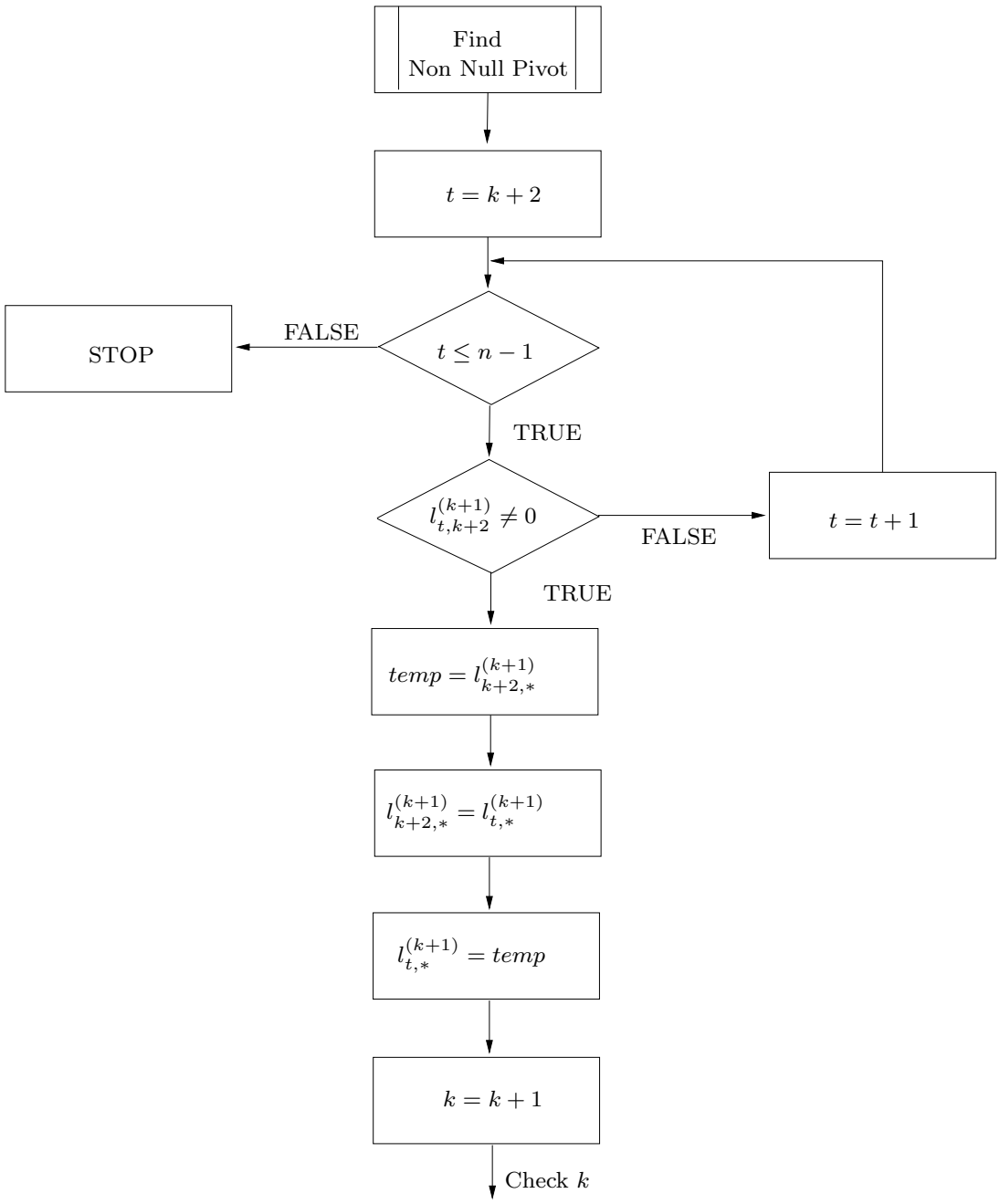


FIGURE C.2 – The subroutine for finding the non null pivot element. The symbol * means "all the column elements"

**Example 9** *Consider the* $(4 \times 3)$ *matrix*

$$L = \begin{pmatrix} l_{11} & l_{12} & l_{13} \\ l_{21} & l_{22} & l_{23} \\ l_{31} & l_{32} & l_{33} \\ l_{41} & l_{42} & l_{43} \end{pmatrix}$$

*with* $L^{(0)} = L$ *and* $l_{0,0}^{[-1]} = 1$. *We have to compute* $k = 3 - 1 = 2$ *matrices* $L^{(1)}, L^{(2)}$ *before completion.*

$k = 0$ *: The first step of the Gaussian-Bareiss elimination method yields :*

$$L^{(1)} = \begin{pmatrix} l_{11} & l_{12} & l_{13} \\ 0 & l_{2,2}^{[1]} & l_{2,3}^{[1]} \\ 0 & l_{3,2}^{[1]} & l_{3,3}^{[1]} \\ 0 & l_{4,2}^{[1]} & l_{4,3}^{[1]} \end{pmatrix}$$

*The first row is kept unchanged and all the components* $l_{i,1}^{[1]}$ *with* $i > 1$ *become zero.*

$k = 1$ *: The second step of the Gaussian-Bareiss elimination method yields :*

$$L^{(2)} = \begin{pmatrix} l_{11} & l_{12} & l_{13} \\ 0 & l_{2,2}^{[1]} & l_{2,3}^{[1]} \\ 0 & 0 & l_{3,3}^{[2]} \\ 0 & 0 & l_{4,3}^{[2]} \end{pmatrix}$$

*The first and the second row are kept unchanged and all the components* $l_{i,2}^{[2]}$ *with* $i > 2$ *become zero.*

**Example 10** *Consider the* $(4 \times 3)$ *matrix*

$$L = \begin{pmatrix} 4 & 1 & 249 \\ 5 & 238 & 7 \\ 250 & 9 & 240 \\ 2 & 243 & 12 \end{pmatrix} \in \mathbb{F}_{251}^{4 \times 3}$$

*with* $L^{(0)} = L$ *and* $l_{0,0}^{[-1]} = 1$. *We have to compute at most* $k = 3 - 1 = 2$ *matrices* $L^{(1)}, L^{(2)}$ *before completion.*

$k = 0$ *: The first step of the Gaussian-Bareiss elimination method yields :*

$$L^{(1)} = \begin{pmatrix} 4 & 1 & 249 \\ 0 & 194 & 38 \\ 0 & 37 & 205 \\ 0 & 217 & 52 \end{pmatrix}$$

*The first row is kept unchanged and all the components* $l_{i,1}^{[1]}$ *with* $i > 1$ *become zero.*

$k = 1$ *: The second step of the Gaussian-Bareiss elimination method yields :*

$$L^{(2)} = \begin{pmatrix} 4 & 1 & 249 \\ 0 & 194 & 38 \\ 0 & 0 & 53 \\ 0 & 0 & 84 \end{pmatrix}$$

**Example 11** *Consider the* $(4 \times 3)$ *matrix*

$$L = \begin{pmatrix} 4 & 1 & 249 \\ 5 & 238 & 7 \\ 50 & 121 & 70 \\ 250 & 9 & 240 \end{pmatrix} \in \mathbb{F}_{251}^{4\times 3}$$

*with* $L^{(0)} = L$ *and* $l_{0,0}^{[-1]} = 1$*. We have to compute at most* $k = 3 - 1 = 2$ *matrices* $L^{(1)}, L^{(2)}$ *before completion.*

$k = 0$ *: The first step of the Gaussian-Bareiss elimination method yields :*

$$L^{(1)} = \begin{pmatrix} 4 & 1 & 249 \\ 0 & 194 & 38 \\ 0 & 183 & 129 \\ 0 & 37 & 205 \end{pmatrix}$$

*The first row is kept unchanged and all the components* $l_{i,1}^{[1]}$ *with* $i > 1$ *become zero.*

$k = 1$ *: The second step of the Gaussian-Bareiss elimination method yields :*

$$L^{(2)} = \begin{pmatrix} 4 & 1 & 249 \\ 0 & 194 & 38 \\ 0 & 0 & 0 \\ 0 & 0 & 53 \end{pmatrix}$$

*It turns out that the* 3*-rd row vanishes. This is due to, according to the remark 22, the fact that the second row of* $L$ *depends linearly on the third row. Indeed,* $l_{2,j}^{(0)} = 10.l_{3,j}^{(0)}$ *mod* 251*.*

*Besides, since* $l_{3,3}^{[2]} = 0$*, according to the Remark 21, it is necessary to switch the* 3*-rd row and the* 4*-th row. We get that :*

$$L^{(2)} = \begin{pmatrix} 4 & 1 & 249 \\ 0 & 194 & 38 \\ 0 & 0 & 53 \\ 0 & 0 & 0 \end{pmatrix}$$

## C.2 Gaussian elimination over $\mathbb{F}_p$

We describe the modification of Gaussian elimination algorithm so that it can work in the finite field $\mathbb{F}_p$.

Consider the matrix $L = (l_{i,j}) \in \mathbb{F}_p^{n \times m}$ with $n > m$. Let $l_{i,j}$ and $l_i$ denote respectively the component at $i$-th row and $j$-th column and the $i$-th row of the matrix $L$.

The Gaussian elimination performs $m$ iterations which involve $m$ columns of the matrix $L$ from the left to the right. The current column which is being computed is called pivot column. The diagonal element in the pivot column is called pivot element. The row corresponded to the pivot element is called pivot row. Each iteration consists of three steps :

i) Ensure that the pivot element has biggest absolute value in the pivot column. If not, exchange the pivot row with one containing the biggest absolute value.

ii) Reduce the pivot element to 1

iii) Eliminate all entries below the pivot element by row elementary operation.

The principal modification takes place in step ii) where the division operation is replaced by multiplying the (multiplicative) inverse over the finite field $\mathbb{F}_p$. To find out the multiplicative inverse, we refers the subsection 3.2.2.3. Consequently, the computation results are always in the finite field $\mathbb{F}_p$. The modified Gauss elimination is described in the following.

**Algorithm 2** Gaussian elimination over $\mathbb{F}_p$

**Input :** $L = (l_{i,j}) \in \mathbb{F}_p^{n \times m}$
**Output :** The upper-triangular form of $L$

Set the current pivot's row $i = 1$
Set the current pivot's column $j = 1$

**while** $j \leq m$ **do**

% Find the biggest element in the current pivot column
$maxi = i$
**for** $k = i + 1$ to $k \leq n$ **do**
**if** $\text{abs}(l_{k,j}) > \text{abs}(l_{maxi,j})$ **then** $maxi = k$
**end if**
**end for**

**if** $l_{maxi,j} \neq 0$ **then**

% Ensure that the pivot element has the biggest absolute value
% in the pivot column
$temp = l_i$
$l_i = l_{maxi}$
$l_{maxi} = temp$

% Reduce the pivot element to 1

% Find the multiplicative inverse of pivot element over the
% finite field $\mathbb{F}_p$
$inv\_pivot =$ multiplicative inverse of $l_{i,j}$

% Multiply $inv\_pivot$ with the pivot row
$l_i = inv\_pivot \times l_i \ (mod\ p)$

% Eliminate all entries below the current pivot element by row
% elementary operation.

**for** $k = i + 1$ to $k \leq n$ **do**
$l_k = l_k - l_{k,j} \times l_i \ (mod\ p)$
**end for**

$i = i + 1$
**end if**

$j = j + 1$
**end while**

**Example 12** *Consider the matrix* $L \in \mathbb{F}_{251}^{8\times 6}$

$$L = \begin{pmatrix} 85 & 0 & 17 & 0 & 0 & 154 \\ 161 & 70 & 12 & 85 & 158 & 149 \\ 41 & 129 & 85 & 161 & 145 & 115 \\ 124 & 243 & 215 & 41 & 59 & 140 \\ 209 & 126 & 38 & 124 & 137 & 121 \\ 106 & 98 & 13 & 209 & 102 & 75 \\ 80 & 166 & 123 & 106 & 136 & 4 \\ 108 & 183 & 187 & 80 & 31 & 103 \end{pmatrix}$$

*Applying the modification of Gaussian elimination algorithm, it takes* 6 *iterations to transform the matrix* $L$ *to the upper-triangular form.*

*At the first iteration, the value of current pivot's row and column are respectively* $i = 1$ *and* $j = 1$. *The biggest element in the first column is found at the* 5*-th row. Thus, we exchange the first row and the* 5*-th row. That yields :*

$$L = \begin{pmatrix} 209 & 126 & 38 & 124 & 137 & 121 \\ 161 & 70 & 12 & 85 & 158 & 149 \\ 41 & 129 & 85 & 161 & 145 & 115 \\ 124 & 243 & 215 & 41 & 59 & 140 \\ 85 & 0 & 17 & 0 & 0 & 154 \\ 106 & 98 & 13 & 209 & 102 & 75 \\ 80 & 166 & 123 & 106 & 136 & 4 \\ 108 & 183 & 187 & 80 & 31 & 103 \end{pmatrix}$$

*The pivot element are* 209. *To reduce the pivot element to 1, we compute the multiplicative inverse of* 209 *( see the Algorithm 1 in subsection 3.2.2.3 for details). We have :*

$$inv\_pivot = 245$$

*Multiplying the pivot row with inv_pivot yields :*

$$L = \begin{pmatrix} 1 & 248 & 23 & 9 & 182 & 27 \\ 161 & 70 & 12 & 85 & 158 & 149 \\ 41 & 129 & 85 & 161 & 145 & 115 \\ 124 & 243 & 215 & 41 & 59 & 140 \\ 85 & 0 & 17 & 0 & 0 & 154 \\ 106 & 98 & 13 & 209 & 102 & 75 \\ 80 & 166 & 123 & 106 & 136 & 4 \\ 108 & 183 & 187 & 80 & 31 & 103 \end{pmatrix}$$

*Applying the row elementary operation to eliminate all entries below the pivot element, we obtain :*

$$L = \begin{pmatrix} 1 & 248 & 23 & 9 & 182 & 27 \\ 0 & 51 & 74 & 142 & 223 & 69 \\ 0 & 1 & 146 & 43 & 213 & 12 \\ 0 & 113 & 124 & 180 & 81 & 55 \\ 0 & 4 & 70 & 239 & 92 & 118 \\ 0 & 165 & 85 & 8 & 137 & 225 \\ 0 & 155 & 40 & 139 & 134 & 103 \\ 0 & 5 & 213 & 112 & 204 & 199 \end{pmatrix}$$

*Keep iterating until the last column and delete all the zero rows, we obtain the upper-triangular form :*

$$L = \begin{pmatrix} 1 & 248 & 23 & 9 & 182 & 27 \\ 0 & 1 & 107 & 140 & 13 & 47 \\ 0 & 0 & 1 & 231 & 44 & 50 \\ 0 & 0 & 0 & 1 & 36 & 118 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

*and the kernel reads :*

$$Ker(L) = [1,\ 0,\ 241,\ 62,\ 0,\ 25]^T$$

# Appendix D

# Daemen's design of self synchronizing stream ciphers

This appendix details the successive versions of self-synchronous stream cipher Moustique : Knot [DGV92], Mosquito [DK05a] and Moustique [DP06].

We recall the updating function :

$$[q_{k+1}^{(j)}]_i = g_K^{(t)}([q_k^{(j-1)}]_i, [q_k^{(j-2)}]_i, \ldots, [q_k^{(1)}]_i, c_k) \ for \ j = 1, \ldots, n \tag{D.1}$$

Each bit of $q_k$ called cell is denoted by

$$[q_k^{(j)}]_i \ 0 \leq i \leq 15$$

and arranged as in FIG. D.1.

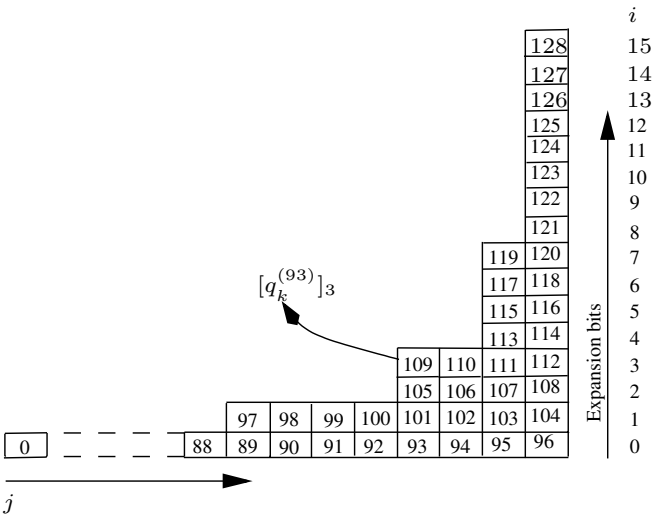


FIGURE D.1 – The arrangement of cells $[q_k^{(j)}]_i$ of the internal state $q_k$

The output function is recalled

$$z_{k+b_s} = s_8(s_7(...(s_0(q_k)))) = h(q_k) \tag{D.2}$$

where the stage $s_i$ $(i = 0, \ldots, 8)$ is detailed in TAB. D.1

| | Knot | | Mosquito | | Moustique | |
|---|---|---|---|---|---|---|
| Stage | Name | Length | Name | Length | Name | Length |
| $s_0$ | CCSR | 128 | Q | 128 | Q | 128 |
| $s_1$ | A | 64 | A | 53 | A | 53 |
| $s_2$ | B | 64 | B | 53 | B | 53 |
| $s_3$ | C | 32 | C | 53 | C | 53 |
| $s_4$ | D | 32 | D | 53 | D | 53 |
| $s_5$ | E | 16 | E | 53 | E | 53 |
| $s_6$ | F | 16 | F | 12 | F | 12 |
| $s_7$ | G | 8 | G | 3 | G | 3 |
| $s_8$ | $z_k$ | 1 | $z_k$ | 1 | $z_k$ | 1 |

TABLE D.1 – The length of shift registers in each stage with respect to the particular algorithm

In the following Sections, we precise all the value $t, i, j$, the updating function $g_K^{(t)}$ and the function $s_i$ $(i = 0, \ldots, b_s - 1)$ corresponding to each particular algorithm. To this end, let $K_j$ $(j = 1, \ldots, 96)$ denotes the $i$-th bit of the secret key $K$. We keep the notation provided in [DGV92], [DK05a], [DP06]. As a result, the XOR operator and AND operator are denoted respectively by + and the concatenation.

# D.1 Knot

**The function $g_K^{(t)}$ in equation** (D.1)
Knot has two updating function $g_K^{(t)}$ (e.g. $t = 1, 2$) which are defined as :

$$\begin{cases} g_K^{(1)} \text{ for } 1 \leq j \leq 96 & \text{and } 0 \leq i \leq 7 \\ g_K^{(2)} \text{ for } j = 96 & \text{and } 8 \leq i \leq 15 \end{cases} \tag{D.3}$$

and

$$g_K^{(1)} = a + b + c(d + 1) + 1$$

$$g_K^{(2)} = a(b + 1) + c(d + 1)$$

The values of $a, b, c, d$ entries are given from TAB. D.2 to TAB. D.6.

**The function $s_i$ in equation** (D.2)
Let $X_l$ denotes the $l$-th bit of the register $X$ in TAB. D.1. All the round transition functions $s_i$ are defined as below :

| $j$ | $i$ | $a$ | $b$ |
|---|---|---|---|
| 89 | $i = 1$ | $[q_k^{(j-1)}]_{i-1}$ | $[q_k^{(71)}]_0$ |
| 93 | $i \geq 2$ | $[q_k^{(j-1)}]_{i-2}$ | $[q_k^{(0)}]_{6j+47}$ |
| 95 | $i \geq 4$ | $[q_k^{(j-1)}]_{i-4}$ | $[q_k^{(0)}]_{6j+11}$ |
| 0 | | $c_k$ | $K_0$ |
| otherwise | | $[q_k^{(j-1)}]_i$ | $K_{j-1}$ |

TABLE D.2 – The value of $a, b$ entries for $1 \leq j \leq 96$ and $0 \leq i \leq 7$

| $i$ | $a$ | $b$ | $c$ | $d$ |
|---|---|---|---|---|
| 8 | $[q_k^{(95)}]_0$ | $[q_k^{(69)}]_0$ | $[q_k^{(94)}]_0$ | $[q_k^{(77)}]_0$ |
| 9 | $[q_k^{(95)}]_1$ | $[q_k^{(70)}]_0$ | $[q_k^{(94)}]_1$ | $[q_k^{(78)}]_0$ |
| 10 | $[q_k^{(95)}]_2$ | $[q_k^{(71)}]_0$ | $[q_k^{(94)}]_2$ | $[q_k^{(79)}]_0$ |
| 11 | $[q_k^{(95)}]_3$ | $[q_k^{(72)}]_0$ | $[q_k^{(94)}]_3$ | $[q_k^{(80)}]_0$ |
| 12 | $[q_k^{(95)}]_4$ | $[q_k^{(73)}]_0$ | $[q_k^{(94)}]_4$ | $[q_k^{(81)}]_0$ |
| 13 | $[q_k^{(95)}]_5$ | $[q_k^{(74)}]_0$ | $[q_k^{(94)}]_5$ | $[q_k^{(82)}]_0$ |
| 14 | $[q_k^{(95)}]_6$ | $[q_k^{(75)}]_0$ | $[q_k^{(94)}]_6$ | $[q_k^{(83)}]_0$ |
| 15 | $[q_k^{(95)}]_7$ | $[q_k^{(76)}]_0$ | $[q_k^{(94)}]_7$ | $[q_k^{(84)}]_0$ |

TABLE D.3 – The value of $a, b, c, d$ entries for $j = 96$ and $8 \leq i \leq 15$

| $j$ | $c$ | $d$ | |
|---|---|---|---|
| $6l + 4$ | $c_k$ | $[q_k^{(6l+2)}]_0$ | $0 \leq l < 16$ |
| $6l + 7$ | $[q_k^{(6l+2)}]_0$ | $c_k$ | $0 \leq l < 15$ |
| $3l + 5$ | $[q_k^{(3l+1)}]_0$ | $[q_k^{(3l+3)}]_0$ | $0 \leq l < 31$ |
| $3l + 6$ | $[q_k^{(3l)}]_0$ | $[q_k^{(3l+4)}]_0$ | $0 \leq l < 30$ |
| $1, 2, 3, 6$ | 0 | 0 | |

TABLE D.4 – The value of $c, d$ entries for $0 \leq j \leq 95$ and $i = 0$

| | | $c$ |
|---|---|---|
| $i = 1$ | $88 < j \leq 92$ | $[q_k^{(j-2)}]_0$ |
| $i = 2, 3$ | $92 < j \leq 94$ | $[q_k^{(j-2)}]_{imod2}$ |
| $0 < i < 8$ | $j = 95$ | $[q_k^{(j-2)}]_{imod4}$ |
| $0 \leq i < 8$ | $j = 96$ | $[q_k^{(j-1)}]_i$ |

TABLE D.5 – The value of $c$ entries for $i > 0$ and $88 < j \leq 96$

| $j$ | $i$ | $d$ | $j$ | $i$ | $d$ | $j$ | $i$ | $d$ |
|---|---|---|---|---|---|---|---|---|
| 89 | 1 | $[q_k^{(81)}]_0$ | 94 | 3 | $[q_k^{(86)}]_0$ | 96 | 0 | $[q_k^{(90)}]_0$ |
| 90 | 1 | $[q_k^{(82)}]_0$ | 95 | 1 | $[q_k^{(87)}]_0$ | 96 | 1 | $[q_k^{(90)}]_1$ |
| 91 | 1 | $[q_k^{(83)}]_0$ | 95 | 2 | $[q_k^{(88)}]_0$ | 96 | 2 | $[q_k^{(91)}]_0$ |
| 92 | 1 | $[q_k^{(84)}]_0$ | 95 | 3 | $[q_k^{(85)}]_0$ | 96 | 3 | $[q_k^{(91)}]_1$ |
| 93 | 1 | $[q_k^{(87)}]_0$ | 95 | 4 | $[q_k^{(92)}]_0$ | 96 | 4 | $[q_k^{(93)}]_1$ |
| 93 | 2 | $[q_k^{(89)}]_0$ | 95 | 5 | $[q_k^{(92)}]_1$ | 96 | 5 | $[q_k^{(93)}]_1$ |
| 93 | 3 | $[q_k^{(89)}]_1$ | 95 | 6 | $[q_k^{(89)}]_0$ | 96 | 6 | $[q_k^{(93)}]_2$ |
| 94 | 1 | $[q_k^{(85)}]_0$ | 95 | 7 | $[q_k^{(89)}]_1$ | 96 | 7 | $[q_k^{(93)}]_3$ |
| 94 | 2 | $[q_k^{(88)}]_0$ | | | | | | |

TABLE D.6 – The value of $d$ entries for $i > 0$ and $88 < j \leq 96$

$$
\begin{aligned}
s_1 &= g_K^{(1)}(CCSR_{6l}, CCSR_{6l+3}, CCSR_{6l+1}, CCSR_{6l+2}) \\
s_2 &= g_K^{(1)}(A_{5l}, A_{5l+3}, A_{5l+1}, A_{5l+2}) \\
s_3 &= g_K^{(1)}(B_{6l}, B_{6l+3}, B_{6l+1}, B_{6l+2}) \\
s_4 &= g_K^{(1)}(C_{5l}, C_{5l+3}, C_{5l+1}, C_{5l+2}) \\
s_5 &= g_K^{(1)}(D_{6l}, D_{6l+3}, D_{6l+1}, D_{6l+2}) \\
s_6 &= g_K^{(1)}(E_{5l}, E_{5l+3}, E_{5l+1}, E_{5l+2}) \\
s_7 &= g_K^{(1)}(F_{6l}, F_{6l+3}, F_{6l+1}, F_{6l+2}) \\
s_8 &= G_0 + G_1(G_2 + 1) + 1
\end{aligned}
$$

## D.2 Mosquito

**The function $g_K^{(t)}$ in equation** (D.1)
Mosquito has three updating function $g_K^{(t)}$ (ie. $t = 1, \ldots, 3$) which are defined as :

$$\begin{cases} g_K^{(1)}([q_k^{(j-1)}]_i, K_{i-1}) & \text{for } 0 \leq j \leq 4 \\ g_K^{(2)}([q_k^{(j-1)}]_i, K_{i-1}, [q_k^{(v)}]_i, [q_k^{(w)}]_i) & \text{for } 4 < j < 96 \quad \text{and } [q_{k+1}^{(96)}]_0 \\ g_K^{(3)}([q_k^{(95)}]_i, [q_k^{(95-i)}]_0, [q_k^{(94)}]_i, [q_k^{(94-i)}]_1) & \text{for } j = 96 \quad \text{and } 1 \leq i \leq 15 \end{cases} \tag{D.4}$$

where :

$$\begin{aligned} g_K^{(1)} &= [q_k^{(j-1)}]_i + K_{i-1} + 1 \\ g_K^{(2)} &= [q_k^{(j-1)}]_i + K_{i-1} + [q_k^{(v)}]_i([q_k^{(w)}]_i + 1) + 1 \text{ and } 0 \leq v, w < j - 1 \\ g_K^{(3)} &= [q_k^{(95)}]_i([q_k^{(95-i)}]_0 + 1) + [q_k^{(94)}]_i([q_k^{(94-i)}]_1 + 1) \end{aligned}$$

The value $w, v$ are given in TAB. D.7 :

| | $v$ | $w$ |
|---|---|---|
| $(i+j)mod3 = 0$ | $i - 4 + (jmod2)$ | $i - 2$ |
| $(i+j)mod3 = 1$ | $i - 6 + (jmod2)$ | $i - 2$ |
| $(i+j)mod6 = 2$ | $i - 5 + (jmod2)$ | 0 |
| $(i+j)mod6 = 5$ | 0 | $i - 2$ |

TABLE D.7 – The value of $w, v$ in the function $g_K^{(2)}$

**The function $s_i$ in equation** (D.2)
Let $X_l$ denotes the $l$-th bit of the register $X$ in TAB. D.1. All the round transition functions $s_i$ are defined as below :

$$\begin{aligned} s_1 &= g_K^{(2)}(Q_{128-l}, Q_{l+18}, Q_{113-l}, Q_{l+1}) && \text{for } 0 \leq l \leq 128 \\ s_2 &= g_K^{(2)}(A_l, A_{l+3}, A_{l+1}, A_{l+2}) && \text{for } 0 \leq l \leq 53 \\ s_3 &= g_K^{(2)}(B_l, B_{l+3}, B_{l+1}, (B_{l+2}) && \text{for } 0 \leq l \leq 53 \\ s_4 &= g_K^{(2)}(C_l, C_{l+3}, C_{l+1}, C_{l+2}) && \text{for } 0 \leq l \leq 53 \\ s_5 &= g_K^{(2)}(D_l, D_{l+3}, D_{l+1}, D_{l+2}) && \text{for } 0 \leq l \leq 53 \\ s_6 &= g_K^{(2)}(E_{4l}, E_{4l+3}, E_{4l+1}, E_{l+2}) && \text{for } 0 \leq l \leq 53 \\ s_7 &= F_{4l} + F_{4l+1} + F_{4l+2} + F_{4l+3} && \text{for } 0 \leq l \leq 12 \\ s_8 &= G_0 + G_1 + G_2 \end{aligned}$$

## D.3 Moustique

**The function $g_K^{(t)}$ in equation** (D.1)
Moustique has four updating functions $g_K^{(t)}$ ($t = 1, \ldots, 4$) which are defined as :

$$[q_{k+1}^{(j)}]_i = g_K^{(t)}([q_k^{(j-1)}]_{imodn_j - 1}, K_{i-1}, [q_k^{(v)}]_{imodn_v}, [q_k^{(w)}]_{imodn_w}) \tag{D.5}$$

where $0 \leq v, w < j-1$ and

$$
\begin{aligned}
g_K^{(1)} &= [q_k^{(j-1)}]_{imodn_{j-1}} + K_{i-1} + 1 && \text{for } 0 \leq j \leq 2 \\
g_K^{(2)} &= [q_k^{(j-1)}]_{imodn_{j-1}} + K_{i-1} + [q_k^{(v)}]_{imodn_v} + [q_k^{(w)}]_{imodn_w} && \text{for } 2 \leq j < 96 \text{ and } [q_{k+1}^{(96)}]_0 \\
g_K^{(3)} &= [q_k^{(j-1)}]_{imodn_{j-1}} + K_{i-1} + [q_k^{(v)}]_{imodn_v}([q_k^{(w)}]_{imodn_w} + 1) + 1 && \text{for } 2 \leq j < 96 \text{ and } [q_{k+1}^{(96)}]_0 \\
g_K^{(4)} &= [q_k^{(95)}]_{imod8}([q_k^{(95-i)}]_0 + 1) + [q_k^{(94)}]_{imod4}([q_k^{(94-i)}]_{1modn_{94-i}} + 1) && \text{for } j = 96 \text{ and } 1 \leq i \leq 15
\end{aligned}
$$

The function $g_K^{(2)}$,$g_K^{(3)}$ and $w, v, n_{j-1}, n_v, n_w$ are chosen as in TAB. D.8 and TAB. D.9 :

| Index | Function | $v$ | $w$ |
|---|---|---|---|
| $(j-i)mod3 = 1$ | $g_2$ | $2(j-i-1)/3$ | $i-2$ |
| $(j-i)mod3 = 2$ | $g_3$ | $j-4$ | $i-2$ |
| $(j-i)mod6 = 3$ | $g_3$ | 0 | $i-2$ |
| $(j-i)mod6 = 0$ | $g_3$ | $j-5$ | 0 |

TABLE D.8 – The value of $w, v$ in the function $g_K^{(2)}$ and $g_K^{(3)}$

| Range $l$ | $n_l$ |
|---|---|
| 1 - 88 | 1 |
| 89 - 92 | 2 |
| 93 - 94 | 4 |
| 95 | 8 |
| 96 | 16 |

TABLE D.9 – The value of $n_{j-1}, n_v, n_w$ in the function $g_K^{(2)}$ and $g_K^{(3)}$

**The function $s_i$ in equation** (D.2)
Let $X_l$ denotes the $l$-th bit of the register $X$ in TAB. D.1. All the round transition functions $s_i$ are defined as below :

$$
\begin{aligned}
s_1 &= g_K^{(2)}(Q_{128-l}, Q_{l+18}, Q_{113-l}, Q_{l+1}) && \text{for } 0 \leq l \leq 128 \\
s_2 &= g_K^{(3)}(A_l, A_{l+3}, A_{l+1}, A_{l+2}) && \text{for } 0 \leq l \leq 53 \\
s_3 &= g_K^{(3)}(B_l, B_{l+3}, B_{l+1}, (B_{l+2}) && \text{for } 0 \leq l \leq 53 \\
s_4 &= g_K^{(3)}(C_l, C_{l+3}, C_{l+1}, C_{l+2}) && \text{for } 0 \leq l \leq 53 \\
s_5 &= g_K^{(3)}(D_l, D_{l+3}, D_{l+1}, D_{l+2}) && \text{for } 0 \leq l \leq 53 \\
s_6 &= g_K^{(3)}(E_{4l}, E_{4l+3}, E_{4l+1}, E_{l+2}) && \text{for } 0 \leq l \leq 53 \\
s_7 &= g_K^{(2)}(F_{4l}, F_{4l+1}, F_{4l+2}, F_{4l+3}) && \text{for } 0 \leq l \leq 12 \\
s_8 &= G_0 + G_1 + G_2
\end{aligned}
$$

# Bibliography


[AL06] G. Alvarez and S. Li. Some basic cryptographic requirements for chaos-based cryptosystems. *Int. J. of Bifurcations and Chaos*, 16(8) :2129–2151, 2006.

[AMB04] F. Anstett, G. Millérioux, and G. Bloch. Global adaptive synchronization based upon polytopic observers. In *Proc. of IEEE International symposium on circuit and systems, ISCAS'04*, pages 728 – 731, Vancouver, Canada, May 2004.

[AMB06] F. Anstett, G. Millérioux, and G. Bloch. Chaotic cryptosystems : Cryptanalysis and identifiability. *IEEE Trans. on Circuits and Systems : Regular papers*, 53(12) :2673–2680, December 2006.

[ASK05] J.M. Amigó, J. Szczepanski, and L. Kocarev. A chaos-based approach to the design of crytographically secure substitutions. *Phys. Lett. A*, 343 :55–60, February 2005.

[Bar68] E. H. Bareiss. Sylvester's identity and multistep integer-preserving gaussian elimination. *Math. Comp.*, 22(103) :565–578, 1968.

[BBBBT04] L. Boutat-Baddas, J. P. Barbot, D. Boutat, and R. Tauleigne. Sliding mode observers and observability singularity in chaotic synchronization. *Mathematical Problems in Engineering*, (1) :11–31, May 2004.

[BBBV02] A. Balluchi, L. Benvenuti, M. D. Di Benedetto, and A. L. Sangiovanni Vincentelli. *Design of Observers for Hybrid Systems*, volume 2289 of *Lecture Notes in Computer Science : Hybrid Systems : Computation and Control*, pages 76–89. Springer-Verlag, Berlin Heidelberg New York, 2002.

[BDR02] M. Boutayeb, M. Darouach, and H. Rafaralahy. Generalized state-space observers for chaotic synchronization and secure communications. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 49(3) :345–349, March 2002.

[BE04] M. Babaali and M. Egerstedt. *Lectures Notes on Hybrid Systems : Computation and Control*, chapter Observability for Switched Linear Systems. Springer-Verlag, Philadelphia, PA, March 2004.

[BGFB94] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15. SIAM Studies in Applied Mathematics, 1994.

[BM65] R. W. Brockett and M. D. Mesarovic. The reproducibility of multivariable systems. *J. Math. Anal. Appl.*, 11 :548–563, July 1965.

[Cal98] J. Calais. *Elenment de theorie des anneaux, anneaux commutative.* Ellipse, 1998.

[CHR00] Huijberts H. J. C., Nijmeijer H., and Willems R. System identification in communication with chaotic systems. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 47(6) :800–808, 2000.

[CP91] T. L Carroll. and L. M. Pecora. Synchronizing chaotic circuits. *IEEE Trans. Circuits and Systems*, 38(4) :453–456, April 1991.

[Dev89] R. L. Devaney. *An introduction to Chaotic Dynamical Systems.* Addison-Wesley, Redwood City, CA, 1989.

[DGV92] J. Daemen, R. Govaerts, and J. Vandewalle. On the design of high speed self-synchronizing stream ciphers. In *Proc. of the ICCS/ISITA'92 conference*, volume 1, pages 279–283, Singapore, November 1992.

[DK02] H. Delfs and H. Knebl. *Introduction to cryptography.* Springer-Verlag, Berlin, 2002.

[DK05a] J. Daemen and Paris K. The self synchronizing stream cipher mosquito : estream documentation, version 2. Technical report, eStream Project, 2005. Available at :www.ecrypt.eu.org/stream/p3ciphers/mosquito/mosquito.pdf.

[DK05b] J. Daemen and P. Kitsos. The self synchronizing stream cipher moustique. *eSTREAM, ECRYPT Stream Cipher Project*, June 2005. Available online at http ://www.ecrypt.eu.org/stream.

[DM02] J. Daafouz and G. Millérioux. Poly-quadratic stability and global chaos synchronization of discrete time hybrid systems. *Special Issue of Mathematics and Computers in Simulation*, 58 :295–307, March 2002.

[DP06] J. Daemen and K. Paris. The self synchronzing stream cipher moustique. Technical report, eStream Project, 2006. Available at : http ://www.ecrypt.eu.org/stream/p3ciphers/mosquito/mosquito_p3.pdf.

[DRI02] J. Daafouz, P. Riedinger, and C. Iung. Stability analysis and control synthesis for switched systems : A switched lyapunov approach. *IEEE Transactions on Automatic Control*, November 2002.

[ET01] Inoue E. and Ushio T. Chaos communication using unknown input observers. *Electronics and communication in Japan part III : Fundamental Electronic Science*, 84(12) :21–27, 2001.

[FLMR95] M. Fliess, J. Levine, P. Martin, and P. Rouchon. Flatness and defect of non-linear systems : introductory theory and examples. *Int. Jour. of Control*, 61(6) :1327–1361, 1995.

[FM97] A. L. Fradkov and A. Y. Markov. Adaptive synchronization of chaotic systems based on speed-gradient method and passification. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 44(10) :905–912, Oct. 1997.

[GM97] G. Grassi and S. Mascolo. Nonlinear observer design to synchronize hyperchaotic systems via a scalar signal. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 44(10) :1011–1014, Oct. 1997.

[GM05] P. Guillot and S. Mesnager. Nonlinearity and security of self-synchronizing stream ciphers. In *Proc. of the 2005 International Symposium on Nonlinear Theory and its Applications (NOLTA 2005)*, Bruges, Belgium, 18-21 October 2005.

[GPO98] Kolumban G., Kennedy M. P., and Chua L. O. The role of synchronization in digital communications using chaos - part II : Chaotic modulation and chaotic synchronization. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 45 :1129–1140, November 1998.

[Has98] M. Hasler. Synchronization of chaotic systems and transmission of information. *International Journal of Bifurcation and Chaos*, 8(4), April 1998.

[Hen76] M. Henon. A two-dimensional map with a strange attractor. *Communicaton of Mathematical Physics*, 50 :69–77, 1976.

[HM97] Dedieu H. and Ogorzalek M. Identification of chaotic systems based on adaptive synchronization. In *Proc. ECCTD'97*, pages 290–295, Budapest, Sept. 1997.

[HNSM91] T. Habutsu, Y. Nishio, I. Sasase, and S. Mori. A secret key cryptosystem by iterating a chaotic map. In *Proc.of EuroCrypt'91, Lecture Notes in Computer Science 0547*, pages 127–140, Berlin, 1991. Springer Verlag.

[HOP92] D. Saupe H. O. Peitgen, H. Juurgen. *Chaos and Fractals : new frontiers of science*. Springer-Verlag, Newyork, 1992.

[HPM93] Dedieu H., Kennedy M. P., and Hasler M. Chaos shift keying : modulation and demodulation of a chaotic carrier using self-synchronizing chua's circuits. *IEEE Trans. Circuits. Syst. II : Anal. Digit. Sign. Process*, 40 :634–642, 1993.

[HPRM04] P. Hawkes, M. Paddon, G. G. Rose, and W. V. Miriam. Primitive specification for sss. Technical report, e-Stream Project, 2004. Available at : http ://www.ecrypt.eu.org/stream/ciphers/sss/sss.pdf.

[HY97] Nijmeijer H. and Mareels I. M. Y. An observer looks at synchronization. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 44 :882–890, October 1997.

[Ike79] K. Ikeda. Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system. *Opt. Commun.*, 30 :257–261, 1979.

[JHFT+05] A. Lj. Juloski, W.P.M.H. Heemels, G. Ferrari-Trecate, R. Vidal, S. Paoletti, and J.H.G. Niessen. Comparision of four procedures for the identification of hybrid systems. In M. Morari and L. Thiele, editors, *In Proc. 8th International Workshop on Hybrid Systems : Computation and Control*, volume 3414, pages 354–369. Springer-Verlag Berlin Heideberg 2005, 2005.

[Knu98] D. E. Knuth. *The Art of Computer Programming, Vol. 2*. Addison-Wesley, Reading, MA, 1998.

[KS04] A. Klimov and A. Shamir. *New cryptographic primitives based on multiword T-functions*, volume 3017, chapter 1, pages 1–15. Springer Berlin / Heidelberg, 2004.

[KSAT06] L. Kocarev, J. Szczepanski, J. M Amigo, and I. Tomosvski. Discrete chaos : part i. *IEEE Trans. on Circuits and Systems I*, 53, 2006.

[KYP00] Lian K-Y. and Liu P. Synchronization with message embedded for generalized lorenz chaotic circuits and its error analysis. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 47(9) :1418–1424, 2000.

[Lan02] S. Lang. *Algebra, Graduate Texts in Mathematics*. Berlin,New York : Springer-Verlag, 2002.

[Lib03] D. Liberzon. *Switching in Systems and Control*. Systems and Control : Foundations and Applications. A Birkhauser, 2003.

[LM91] X. Lai and J. M. Massey. A proposal for a new block encryption standard. In *Advances in Cryptology (EUROCRYPT'90)*, Lectures Notes in Computer Science 473, pages 386–404, Aarhus, Denmark, 1991. Springer-Verlag.

[LM99] D. Liberzon and A. S. Morse. Basic problems in stability and design of switched system. *IEEE Control Systems*, 19(5) :59–70, October 1999.

[LY75] T-Y. Li and J. A. Yorke. Period three implies chaos. *Amer. Math. Monthly*, 82 :985–992, 1975.

[MAD08] G. Millérioux, J. M. Amigó, and J. Daafouz. A connection between chaotic and conventional cryptography. *IEEE Trans. on Circuits and Systems I : Regular Papers*, 55(6), July 2008.

[Mas92] J.L. Massey. *Contemporary cryptology : an introduction*. G.J. Simmons, New York, ieee press edition, 1992.

[Mat89] R. Matthews. On the derivation of a chaotic encryption algorithm. *Cryptologia*, 13 :29–41, 1989.

[Mau91] U. M. Maurer. New approaches to the design of self-synchronizing stream cipher. *Advance in Cryptography, In Proc. Eurocrypt '91, Lecture Notes in Computer Science*, pages 548–471, 1991.

[May76] R.M. May. Simple mathematical models with complicated dynamics. *Nature*, 261 :459–470, 1976.

[MD03] G. Millérioux and J. Daafouz. An observer-based approach for input independent global chaos synchronization of discrete-time switched systems. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 50(10) :1270–1279, October 2003.

[MD04] G. Millérioux and J. Daafouz. Unknown input observers for message-embedded chaos synchronization of discrete-time systems. *International Journal of Bifurcation and Chaos*, 14(4) :1357–1368, April 2004.

[MD06] G. Millérioux and J. Daafouz. *Chaos in Automatic Control*, chapter Polytopic observers for synchronization of chaotic maps, pages 323–344. Control Engineering Series. CRC Press, 2006.

[MD07] G. Millérioux and J. Daafouz. Invertibility and flatness of switched linear discrete-time systems. In *Proc. of the 10th International Conference on Hybrid Systems : Computation and Control (HSCC'07)*, volume 44, pages 714–717, Pisa, Italy, April 2007. Springer Verlag.

[MD09] G. Millérioux and J. Daafouz. Flatness of switched linear discrete-time systems. *IEEE Trans. on Automatic Control*, 54(3) :615–619, March 2009.

[Mil97] G. Millérioux. Chaotic synchronization conditions based on control theory for systems described by discrete piecewise linear maps. *International Journal of Bifurcation and Chaos*, 7(7) :1635–1649, 1997.

[MM98] G. Millérioux and C. Mira. Coding scheme based on chaos synchronization from noninvertible maps. *International Journal of Bifurcation and Chaos*, 8(10) :2019–2029, 1998.

[Mon85] P. Montgomery. Modular multiplication without trial division. *Mathematics of Computation, American Mathematical Society, Providence, Rhode Island*, 44 :519–521, April 1985.

[MOV96] A. J. Menezes, P. C. Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, October 1996.

[MVH93] Cuomo K. M., Oppenheim A. V., and Strogatz S. H. Synchronization of lorenz-based chaotic circuits with applications to communications. *IEEE Trans. Circuits. Syst. II : Anal. Digit. Sign. Process*, 40(10) :626–633, 1993.

[MWO97] Itoh M., Wu C. W., and Chua L. O. Communications systems via chaotic signals from a reconstruction viewpoint. *International Journal of Bifurcation and Chaos*, 7(2) :275–286, 1997.

[Ogo93] M. J. Ogorzalek. Taming chaos - part I : synchronization. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 40(10) :693–699, 1993.

[PM01] Palaniyandi P. and Lakshmanan M. Secure digital signal transmission by multistep parameter modumation and alternative driving

of transmitter variables. *International Journal of Bifurcation and Chaos*, 11(7) :2031–2036, 2001.

[SAMK05] J. Szczepanski, J.M. Amigó, T. Michalek, and L. Kocarev. Cryptographically secure substitutions based on the approximation of mixing maps. *IEEE Trans. Circuits and Systems I : Regular Papers*, 52(2) :443–453, February 2005.

[Sch01] R. Schmitz. Use of chaotic dynamical systems in cryptography. *Journal of the Franklin Institute*, 338 :429–441, 2001.

[SH06] S. Sundaram and C. Hadjicostis. Designing stable inverters and state observers for switched linear systems with unknown inputs. In *Proc. of the 45th IEEE Conference on Decision and Control*, San Diego, CA, USA, December 2006.

[Sha49] C. E. Shannon. Communication theory of secrecy systems. *Bell Systems Tech. Journ.*, 28 :657–715, 1949.

[Sil69] L. M. Silverman. Inversion of multivariable linear systems. *IEEE Trans. Automatic Control*, 14(3) :270–276, June 1969.

[SM69] M. K. Sain and J. L. Massey. Invertibility of linear time-invariant dynamical systems. *IEEE Trans. Automatic Control*, 14 :141–149, 1969.

[SRA04] H. Sira-Ramirez and S. K. Agrawal. *Differentially Flat Systems*. Marcel Dekker, New York, 2004.

[TL08] A. Tanwani and D. Liberzon. Invertibility of nonlinear switched systems. In *In. Proc. 47th IEEE Conference on Decision and Control, CDC 2008*, Cancun,Mexico, 2008.

[UMW96] Feldmann U., Hasler M., and Schwarz W. Communication by chaotic signals :the inverse system approach. *Int. J. of Circuit Theory Appl.*, 24 :551–579, 1996.

[UOL+93] Parlitz U., Chua L. O., Kocarev L., Halle K. S., and Shang A. Transmission of digital signals by chaotic synchronization. *International Journal of Bifurcation and Chaos*, 3(2) :973–977, 1993.

[VL08] L. Vu and D. Liberzon. Invertibility of switched linear systems. *Automatica*, 44(4) :949–958, 2008.

[VMS03] R. Vidal, Y. Ma, and S. Sastry. An algebraic geometric approach to the identification of a class of linear hybrid systems. *42nd IEEE Conference on Decision and Control 2003 (CDC'03)*, 2003.

[Wan91] D. Wang. Elimination theory, methods and practice. *Mathematics and Mathematics-Mechanization*, pages 91–137, 1991. available at http ://www-calfor.lip6.fr/ wang/.

[WM76] Diffie W. and Hellman M. New directions in cryptography. *IEEE Trans. on Information Theory*, 22(6) :644–654, 1976.

[WO93] Wu C. W. and Chua L. O. A simple way to synchronize chaotic systems with applications to secure communications systems. *International Journal of Bifurcation and Chaos*, 3(6) :1619–1627, 1993.

[Yan04] T. Yang. A survey of chaotic secure communication systems. *Int. J. of Computational Cognition*, 2004. (available at `http://www.YangSky.com/yangijcc.htm`).

[ZP02] Jiang Z-P. A note on chaotic secure communication systems. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 49(1) :92–96, January 2002.

Nancy-Université INPL

**École doctorale IAEM Lorraine**
**DFD Automatique et Production Automatisée**
**Institut National Polytechnique de Lorraine**

# Théorie du contrôle et systèmes hybrides dans un contexte cryptographique

# RÉSUMÉ ÉTENDU DE LA THÈSE

présentée et soutenue publiquement le 12/11/2009

pour l'obtention du

**Doctorat de l'Institut National Polytechnique de Lorraine**

**Spécialité Automatique et Traitement du signal**

par

Phuoc VO TAN

**Composition du jury**

| | | |
|---|---|---|
| *Président :* | Mohammed MSAAD | Professeur<br>Ensicaen, Caen |
| *Rapporteurs :* | Nacim RAMDANI | Professeur<br>Université Montpellier 2, Montpellier |
| | Krishna BUSAWON | Professeur, NCRLab<br>Northumbria University, Royaume-Uni |
| *Examinateur :* | Philippe GUILLOT | Maître de Conférences<br>Université Paris 8, Paris |
| *Directeur de thèse :* | Gilles MILLERIOUX | Professeur<br>Université Henri Poincaré - Nancy 1, Nancy |
| *Co-directeur de thèse :* | Jamal DAAFOUZ | Professeur<br>Institut National Polytechnique de Lorraine, Nancy |

CRAN

**Centre de Recherche en Automatique de Nancy**
**UMR 7039 Nancy-Université – CNRS**
**2, avenue de la forêt de Haye 54516 Vandœuvre-lès-Nancy**
**Tél. +33 (0)3 83 59 59 59 Fax +33 (0)3 83 59 56 44**

Mis en page avec la classe thloria.

# Table des matières

# Introduction

La thèse traite de l'utilisation des systèmes hybrides dans le contexte particulier des communications sécurisées et de la cryptographie. Ce travail est motivé par les faits suivants. L'essor considérable des communications qui a marqué ces dernières décennies nécessite des besoins croissants en terme de sécurité des échanges et de protection de l'information. Dans ce contexte, la cryptographie joue un rôle central puisque les informations transitent la plupart du temps au travers de canaux publics. Parmi les nombreuses techniques de chiffrement existants, le chiffrement par flot se distingue tout particulièrement lorsqu'on le débit d'une communication sécurisée est privilégié. Les chiffreurs par flot sont construits à partir de générateurs de séquences complexes décrits par des systèmes dynamiques et devant être synchronisés de part et d'autre du canal d'échanges. Les objectifs et les résultats de ce travail se déclinent en trois points. Tout d'abord, l'intérêt d'utiliser des systèmes hybrides en tant que primitives cryptographiques est motivé. Par la suite, une étude comparative est menée afin d'établir une connexion entre les algorithmes de masquage de l'information basés sur le chaos et les algorithmes de chiffrement usuels. L'étude porte exclusivement sur des considérations structurelles et repose sur des concepts de la théorie du contrôle, en particulier l'inversibilité à gauche et la platitude. On montre que la technique de masquage dite par inclusion, qui consiste à injecter l'information à protéger dans une dynamique complexe, est la plus efficace. De plus, on montre que sous la condition de platitude, un système de masquage par inclusion est structurellement équivalent à un chiffreur par flot particulier appelé auto-synchronisant. Enfin, des méthodes de cryptanalyse pour évaluer la sécurité du masquage par inclusion sont proposées pour une classe particulières de systèmes hybrides à savoir les systèmes linéaires à commutations. A nouveau, des concepts de la théorie du contrôle sont utilisés, il s'agit de l'identifiabilité paramétrique et des algorithmes d'identification. Des spécificités relatives au contexte particulier de la cryptographie sont prises en compte. En effet, contrairement à la plupart des cas rencontrés dans le domaine du contrôle où les variables des modèles dynamiques sont continues car relatives à des systèmes physiques, les variables prennent ici des valeurs discrètes. Les modèles dynamiques sont en effet décrits non plus dans le corps des réels mais dans des corps finis en vue d'une implémentation sur des machines à états finis tels ordinateur ou tout autre dispositif numérique.

# Chapitre 1

# Chiffrement basé sur le chaos

Pendant longtemps, le chaos a été considéré comme "dangereux" ou indésirable par la communauté scientifique. Cependant, dans les années 90, des scientifiques ont réalisé que le chaos pouvait être contrôlé et ont commencé à chercher ses applications possibles. Les signaux issus des systèmes chaotiques sont imprédictibles à long terme, peuvent présenter des propriétés spectrales et statistiques proches de l'aléatoire (signaux à large spectre, autocorrélation réduite), bien qu'issus de systèmes déterministes. Ces caractéristiques sont liées aux propriétés requises par les schémas de chiffrement, telles que la confusion et la diffusion de Shannon, usuellement rencontrées en cryptographie [BRV05]. En 1990, [PC90] ont montré que les systèmes chaotiques peuvent être synchronisés. Une des applications du chaos qui a alors intéressé les chercheurs est l'utilisation de systèmes chaotiques à des fins de chiffrement. De nombreux schémas de chiffrement basés sur le chaos ont été proposés dans la littérature. En revanche, très peu de travaux ont réellement fait un lien entre les algorithmes de chiffrement standard et ceux basés sur la génération de séquences chaotiques. On notera cependant des premiers travaux comparatifs dans [GKS97] [DKVS98] [Koc01].

Ce chapitre est organisé comme suit. La Section 1.1 est une introduction à la notion synchronisation du chaos. Puis, les Sections suivantes sont consacrées aux différents schémas de chiffrement par le chaos rencontrés dans la littérature, le masquage additif (Section 1.2), la modulation chaotique (Section 1.3), la modulation paramétrique (Section 1.4) et le chiffrement par inclusion (Section 1.5).

## 1.1 Synchronisation du chaos

Le principe des schémas de chiffrement basé sur le chaos consiste à mélanger l'information $m_k$ avec une séquence chaotique issue d'un émetteur, décrit généralement par une représentation d'état avec le vecteur d'état $x_k$. Seule la sortie $y_k$ de l'émetteur est transmise au récepteur. Le récepteur a pour rôle d'extraire l'information originale du signal reçu $y_k$. La récupération de l'information est généralement basée sur la synchronisation des états $x_k$ de l'émetteur et des états

$\hat{x}_k$ du récepteur, c'est-à-dire :

$$\lim_{k\to\infty} ||x_k - \hat{x}_k|| = 0, \quad \forall \hat{x}_0 \tag{1.1}$$

ou :

$$\exists k_f, \quad ||x_k - \hat{x}_k|| = 0, \quad \forall k > k_f, \quad \forall \hat{x}_0 \tag{1.2}$$

La relation (1.1) correspond à une convergence asymptotique et (1.2) correspond à une convergence en un nombre fini d'itérations.

Différentes techniques d'injection de l'information dans un système chaotique ont été proposées dans la littérature, telles que le masquage additif [MVH93b], la modulation chaotique [HPM93], la modulation paramétrique [UOL+93], l'approche par inclusion. Un aperçu de ces techniques peut être trouvé dans [Has98] [Yan04], aussi bien pour les systèmes à temps discret que pour les systèmes à temps continu. Dans ce chapitre, ces différentes techniques sont présentées dans le cas des systèmes à temps discret.

## 1.2 Masquage additif

Le masquage additif est lié aux travaux de[WO93] et de [MVH93a], initialement effectués pour les systèmes à temps continu. Le principe de ce schéma consiste à effectuer une simple addition entre le signal de sortie de l'émetteur $y_k$ et l'information $m_k$. L'émetteur (générateur de chaos) et le récepteur ont pour représentation d'état, respectivement :

$$\begin{cases} x_{k+1} = f(x_k) \\ y_k \quad = h(x_k) + m_k \end{cases} \tag{1.3}$$

$$\begin{cases} \hat{x}_{k+1} \quad = f(\hat{x}_k), \\ \hat{y}_k \quad = h(\hat{x}_k). \end{cases} \tag{1.4}$$

où $x_k$ (resp. $\hat{x}_k$) $\in \mathbb{R}^n$ est le vecteur d'état de l'émetteur (resp. du récepteur), $y_k \in \mathbb{R}$ (resp. $\hat{y}_k$) la sortie de l'émetteur (resp. du récepteur), $m_k \in \mathbb{R}$ l'information à masquer. La FIG. 1.1 illustre ce mode de masquage.

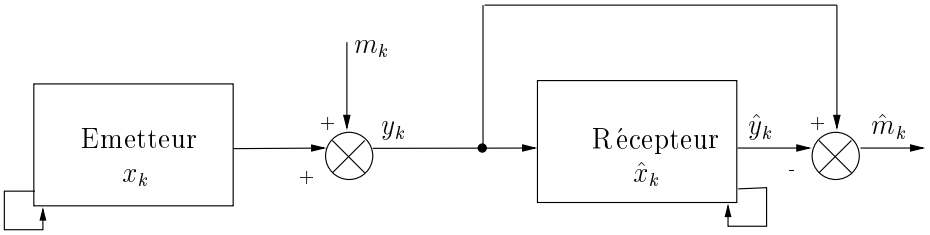


FIGURE 1.1 – Masquage additif

La reconstruction de l'information nécessite la synchronisation (1.1) ou (1.2) de l'émetteur et du récepteur. L'information est alors récupérée en soustrayant la sortie du récepteur avec celle de l'émetteur :

$$\hat{m}_k = y_k - \hat{y}_k \tag{1.5}$$

Le principal inconvénient de cette approche est que l'information jouant le rôle d'une perturbation qui ne peut pas être en général totalement rejetée, la synchronisation n'est jamais exacte, même si l'amplitude du signal information est faible par rapport à celle du signal de sortie. Par conséquent, cette méthode n'est pas satisfaisante.

## 1.3 Modulation chaotique

La modulation chaotique, due à [HPM93], est aussi connue sous le nom de "chaos shift-keying" ou "chaotic switching", en anglais.

Côté émetteur, à chaque symbole $m_k = m_i$ de l'information, appartenant à un ensemble fini $\{m_1, \ldots, m_N\}$, correspond un signal $y_k$ issu d'un système chaotique décrit par :

$$\begin{cases} x_{k+1} = f_i(x_k) \\ y_k \quad\; = h_i(x_k) \end{cases} \tag{1.6}$$

où $i \in \{1, \ldots, N\}$, $x_k \in \mathbb{R}^n$ est le vecteur d'état, $y_k \in \mathbb{R}$ la sortie. Le cas le plus simple correspond à une information binaire. Dans ce cas, seulement deux systèmes émetteur (1.6), avec $i = \{1, 2\}$, sont nécessaires, l'un correspondant à $m_1 = 0$ et l'autre à $m_2 = 1$.

Ce schéma de modulation est représenté sur la FIG. 1.2.

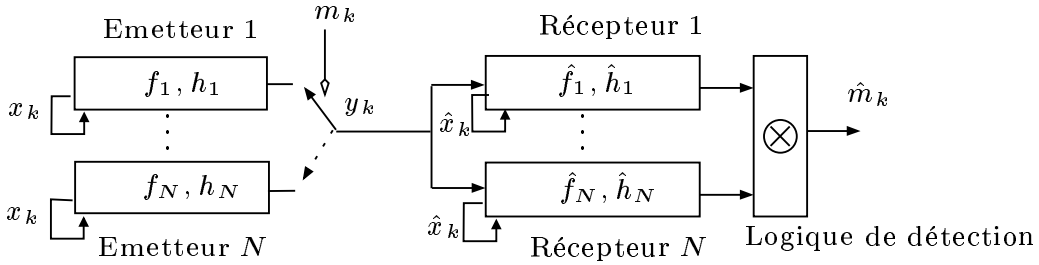


FIGURE 1.2 – Modulation chaotique

Le rôle du récepteur est de détecter quel émetteur a produit la sortie $y_k$. Pour

cela, le récepteur est composé d'autant de systèmes que l'émetteur, décrits par :

$$\begin{cases} \hat{x}_{k+1} = \hat{f}_i(\hat{x}_k) \\ \hat{y}_k \quad = \hat{h}_i(\hat{x}_k) \end{cases}, \quad i = 1, \ldots, N \tag{1.7}$$

Il existe deux méthodes de détection, l'une cohérente et l'autre non cohérente. La détection non cohérente utilise des approches statistiques basées principalement sur l'analyse de la corrélation entre $y_k$ et $\hat{y}_k$.

La détection cohérente nécessite la synchronisation de l'émetteur et du récepteur. Seulement un des $N$ récepteurs peut se synchroniser selon la valeur courante $m_i$. Une logique de détection permet alors de reconstruire l'information, en analysant les erreurs de reconstruction $y_k - \hat{y}_k$ associées à chaque récepteur. Un banc d'observateurs peut jouer le rôle de récepteur, en garantissant la convergence à zéro de l'erreur de reconstruction d'état.

## 1.4 Modulation paramétrique

La modulation paramétrique consiste à moduler un ou plusieurs paramètres du générateur de chaos par l'information $m_k$. Il en résulte un "mélange" multiplicatif entre le ou les paramètres du générateur de chaos et l'information.

Quand l'information prend un nombre fini de valeurs, on parle de modulation discrète. Le cas le plus simple correspond à une information binaire $m_k$ [HPM93] [UOL+93] [CH00], où un "1" est codé en transmettant un signal chaotique et où un "0" est codé en transmettant un autre signal chaotique, mais peut être étendu à un cas plus général [PM01]. La FIG. 1.3 illustre ce schéma de masquage. Le

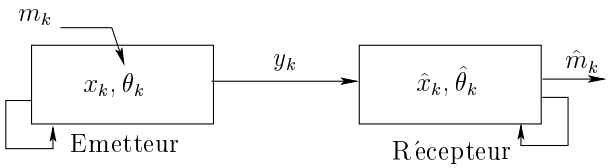


FIGURE 1.3 – Modulation paramétrique

système émetteur peut être décrit par la représentation d'état :

$$\begin{cases} x_{k+1} = f(x_k, \theta_k) \\ y_k \quad = h(x_k) \end{cases} \tag{1.8}$$

où $x_k \in \mathbb{R}^n$ est le vecteur d'état, $y_k \in \mathbb{R}$ la sortie, $\theta_k \in \mathbb{R}^l$ le vecteur des paramètres modulés. La fonction $f$ est non linéaire et les fonctions $g$ et $h$ sont linéaires. Le paramètre $\theta_k$ varie dans le temps car il est modulé par l'information $m_k$. Quand $m_k = m_i$, $\theta_k = \theta_i$.

La modulation discrète peut être étendue au cas où l'information $m_k$ ne prend plus un nombre fini de valeurs. On parle alors de modulation continue.

Pour la partie émetteur, le principe reste identique à celui de la modulation discrète. Toute la problématique se situe, côté récepteur, dans la récupération des paramètres. En effet, le récepteur doit se synchroniser sur l'émetteur. La récupération des paramètres modulés peut alors se baser sur l'estimation simultanée état/paramètre, via un observateur adaptatif.

## 1.5 Chiffrement par inclusion

Dans ce schéma, le signal information est injecté dans un générateur de chaos jouant le rôle d'émetteur, qui admet la représentation d'état suivante :

$$\begin{cases} x_{k+1} = f_\theta(x_k, m_k) \\ y_k \quad\; = h_\theta(x_k, m_k) \end{cases} \tag{1.9}$$

où $x_k \in \mathbb{R}^n$ est le vecteur d'état, $y_k \in \mathbb{R}$ la sortie, $m_k \in \mathbb{R}$ l'information à masquer, $\theta = [\theta_1, \ldots, \theta_l] \in \Theta \subset \mathbb{R}^l$ le vecteur de paramètres constants du système chaotique. Chaque symbole $m_k$ est injecté dans la dynamique non linéaire chaotique $f_\theta$, paramétrée par $\theta$, générant $x_k$. Seule la sortie $y_k$ est transmise au récepteur, $x_k$ étant un vecteur d'état interne qui n'est pas accessible. Ce schéma est représenté sur la FIG.1.4.

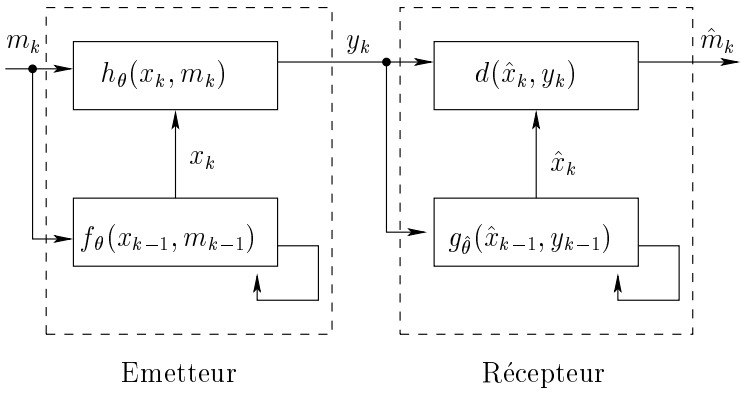


FIGURE 1.4 – Chiffrement par inclusion

Le récepteur a pour représentation d'état générale :

$$\begin{cases} \hat{x}_{k+1} = g_{\hat{\theta}}(\hat{x}_k, y_k) \\ \hat{m}_k \quad\; = d(\hat{x}_k, y_k) \end{cases} \tag{1.10}$$

Le récepteur doit être synthétisé de telle sorte que l'information $m_k$ puisse être reconstruite avec pour seule donnée la sortie de l'émetteur $y_k$. En effet, l'état interne $x_k$ n'est pas directement transmis au récepteur, mais est nécessaire pour

reconstruire l'information. La fonction $g_{\hat{\theta}}$ est choisie de telle sorte que si $\hat{\theta} = \theta$, alors $\hat{x}_k = x_k$, quel que soit $\hat{x}_0$ et indépendamment de l'information $m_k$ qui joue le rôle d'une entrée externe au système dynamique, soit, en considérant une convergence asymptotique :

$$\lim_{k\to\infty} ||x_k - \hat{x}_k|| = 0, \quad \forall m_k, \quad \forall \hat{x}_0 \tag{1.11}$$

soit, en considérant une convergence en un nombre fini d'itérations :

$$\exists k_f, \quad ||x_k - \hat{x}_k|| = 0, \quad \forall k > k_f, \quad \forall m_k, \quad \forall \hat{x}_0 \tag{1.12}$$

Lorsque (1.11) ou (1.12) est vérifiée, on dit que l'émetteur et le récepteur se synchronisent globalement, indépendamment de l'entrée (**I**nput **I**ndependent **G**lobal **S**ynchronization, IIGS) [MD04].

La synchronisation de $x_k$ et de $\hat{x}_k$ est formulée comme un problème d'inversion de systèmes dans [UMW96] et comme un problème d'observation à entrées inconnues dans [TP99] pour les systèmes de Lur'e, dans [MD03] [MBA$^+$03] pour les systèmes linéaires par morceaux et dans [MD04] pour les systèmes admettant une forme polytopique.

## 1.6 Conclusion

Nous avons décrit des schémas de chiffrement basés sur le chaos, le masquage additif, la modulation chaotique, la modulation paramétrique et le chiffrement par inclusion. Dans le masquage additive, la synchronisation est imparfaite. Dans la modulation chaotique, il s'agit d'un observateur adaptatif pour reconstruire l'information. La synchronisation donc nécessite une transitoire et est assurée quand le paramètre du system varie lentement. Dans le chapitre suivant, nous nous intéressons plus particulièrement au chiffrement par inclusion car il permet de pallier les différents problèmes rencontrés ci-dessus.

# Chapitre 2

# Quelques notions essentielles de la théorie du contrôle

Ce chapitre introduit les notions caractérisant un système dynamique qui joue un rôle fondamental en vue d'établir une connexion entre les techniques de "brouillage" et les algorithmes de chiffrement par flot. Ce chapitre est organisé comme suit. La Section 2.1 rappelle les notions essentielles du contrôle. Elles sont appelées le degré relatif, l'invertibilité et la platitude. La Section 2.2 introduit l'identification sur les corps finis. Nous nous intéressons particulièrement aux systèmes linéaires à commutation qui sont la dynamique du schéma chiffrement par inclusion présenté au Chapitre 1. La possibilité de reconstruction des paramètres de cette dynamique qui sont la clé secrète est aussi abordée. Ce problème est directement lié au concept d'identifiabilité paramétrique. Notons que les variables du système dynamique ont les valeurs continues ($\mathbb{R}^n$) car elles représentent les quantités physiques. Quand l'implémentation digitale est envisagée, toutes les variables doivent être numérisées et appartiennent donc au corps fini ($\mathbb{F}_p$). Cependant, la performance en débit sera très limitée. Pour cette raison, nous nous intéressons aux systèmes en temps discrète dont les variables sont liées directement au corps fini ($\mathbb{F}_p$).

## 2.1 Rappel sur la théorie du contrôle

Considérons le système linéaire mono-entrée mono-sortie à commutation suivant :

$$\left\{ \begin{array}{lcl} x_{k+1} & = & A_{\sigma(k)}x_k + B_{\sigma(k)}m_k \\ y_k & = & C_{\sigma(k)}x_k + D_{\sigma(k)}m_k \end{array} \right. \tag{2.1}$$

où $m_k \in \mathbb{F}_p$, $y_k \in \mathbb{F}_p$ et $x_k \in \mathbb{F}_p^n$. La fonction de commutation $\sigma(k)$ est donnée par

$$\sigma : k \in \mathbb{N} \mapsto j = \sigma(k) \in \{1, \ldots, J\}$$

Toutes les matrices, appelées $A_{\sigma(k)} \in \mathbb{F}_p^{n\times n}$, $B_{\sigma(k)} \in \mathbb{F}_p^{n\times 1}$, $C_{\sigma(k)} \in \mathbb{F}_p^{1\times n}$ et $D_{\sigma(k)} \in \mathbb{F}_p$ appartiennent respectivement aux ensembles finis $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ et $(D_j)_{1\leq j\leq J}$. A l'instant $k$, l'indice $j$ correspond à un sous système déterminé

par la fonction de commutation $\sigma_t$.

Trois propriétés caractérisant un système dynamique jouent un rôle fondamental en vue d'établir une connexion entre les techniques de "brouillage" et les algorithmes de chiffrement par flot. Elle sont rappelées ci-dessous et peuvent être trouvées dans [Isi95][FM01].

**Définition 1** Le **degré relatif** d'un système dynamique est le nombre minimum $r$ d'itérations de la sortie $y_k$ requises afin que la sortie $y_{k+r}$ à l'instant $k+r$ dépende de manière explicite de l'entrée $m_k$. Lorsqu'un système possède un degré relatif fini ($r < \infty$), il existe une fonction notée $h^{(r)}$ qui lie la paire $(x_k, m_k)$ à $y_{k+r}$

$$y_{k+r} = h^{(r)}(x_k, m_k) \tag{2.2}$$

et qui vérifie l'implication suivante :

$$\exists x_k,\ \exists m_k,\ \exists m'_k \text{ tq } m_k \neq m'_k \Rightarrow h^{(r)}(x_k, m_k) \neq h^{(r)}(x_k, m'_k)$$

En itérerant (2.1), on obtient l'expression de $y_{k+i}$ :

$$y_{k+i} = C_{\sigma(k+i)} A_{\sigma(k)}^{\sigma(k+i-1)} x_k + \sum_{j=0}^{j=i} \mathcal{T}_{\sigma(k)}^{i,j} u_{k+j} \tag{2.3}$$

avec

$$\mathcal{T}_{\sigma(k)}^{i,j} = C_{\sigma(k+i)} A_{\sigma(k+j+1)}^{\sigma(k+i-1)} B_{\sigma(k+j)} \text{ si } j \leq i-1, \quad \mathcal{T}_{\sigma(k)}^{i,i} = D_{\sigma(k+i)} \tag{2.4}$$

et la matrice de transition est donnée par :

$$\begin{array}{rcll} A_{\sigma(k_0)}^{\sigma(k_1)} & = & A_{\sigma(k_1)} A_{\sigma(k_1-1)} \ldots A_{\sigma(k_0)} & \text{si } \ k_1 \geq k_0 \\ & = & \mathbf{1}_n & \text{si } \ k_1 < k_0 \end{array}$$

En accordant avec la Définition 2.1, le degré relatif $r$ à l'équation (2.1) est donnée par :

- $r = 0$ si $\mathcal{T}_{\sigma(k)}^{0,0} \neq 0$ pour tout $k$
- pout tout $k$, il existe un plus petit entier $r < \infty$, tel que

$$\begin{array}{l} \mathcal{T}_{\sigma(k)}^{i,j} = 0 \text{ for } \ i = 0, \ldots, r-1 \ \text{ and } \ j = 0, \ldots, i \\ \mathcal{T}_{\sigma(k)}^{r,0} \neq 0 \end{array} \tag{2.5}$$

Quand (2.1) possède un degré relatif $r$, sa sortie à l'instant $k+r$ est donnée par :

$$y_{k+r} = C_{\sigma(k+r)} A_{\sigma(k)}^{\sigma(k+r-1)} x_k + \mathcal{T}_{\sigma(k)}^{r,0} u_k \tag{2.6}$$

**Définition 2** Un système dynamique est dit **inversible à gauche** si l'entrée $m_k$ peut être déterminée de manière unique à partir d'un nombre fini de sorties retardées $y_{k+i}$ $(i = 1, \ldots, R)$ pour toute condition intiale $x_k$.

Pour une introduction à la platitude on peut se référer à [FLMR95].

**Définition 3** Une sortie $y_k$ d'un système dynamique est dite **plate** si toutes les autres variables du système peuvent être exprimées au travers de fonctions dépendant uniquement des retards ou avances de $y_k$. En particulier il existe une fonction notée $\mathcal{F}$ et deux entiers relatifs $t_1$ et $t_2$ tels que :

$$x_k = \mathcal{F}(y_{k+t_1}, \cdots, y_{k+t_2}) \tag{2.7}$$

## 2.2 Identification sur les corps finis

Soit $\theta$ un vecteur paramètre qui se compose d'un sous-ensemble d'élément de $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ et $(D_j)_{1\leq j\leq J}$ de l'équations (2.1). Ce vecteur jouera le rôle de la clé secrète, nous devons donc introduire d'une procédure d'identification de $\theta$.

Le principe est basé sur la relation entrée-sortie de (2.1.) Notons que les systèmes à commutation obéissant d'une sortie plate sont les candidates uniques de transmetteur. Quand le système (2.1) est plat, il est facile d'obtenir systématiquement sa relation entrée-sortie. En effet, si (2.1) est plat avec sa sortie plate $y_k$, le vecteur d'état $x_k$ obéit à l'équation (2.7). En remplaçant l'expression de $x_k$ (2.7) à l'équation (2.6), on peut déduire la relation entrée-sortie :

$$y_{k+r} = \sum_{j=0}^{K-1} a_j(\sigma_t) y_{k+j+r-K} + c(\sigma_t) u_k \tag{2.8}$$

où $c(\sigma_t)$ et $a_j(\sigma_t)$'s $(j = 0, \ldots, K-1)$ sont les coefficients dépendant des séquences de la fonction de commutation $\sigma_t$ $(t = 1, \ldots, N)$ et des composants des matrices $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ et $(D_j)_{1\leq j\leq J}$ dans l'équations (2.1)

**Proposition 1** *Le nombre maximum $N = N_{max}$ des relations entrée-sortie indépendamment de la variation de $J$ est $N_{I/O} = p^{K+1}$*

**Preuve 1** Le preuve est déduite à partir de deux considérations suivantes. La relation entrée-sortie (2.8) implique $K + 1$ coefficients. Chacun appartient au corps fini $\mathbb{F}_p$ qui se compose d'un nombre fini d'éléments $p$.

Basé sur (2.8), deux procedures peuvent être envisagées selon l'accessibilité de $\sigma_t$.

$\sigma_t$ est accessible
Pour chaque $\sigma_t$, les paramètres $c(\sigma_t)$ et $a_j(\sigma_t)$ sont linéaires dans la relation entrée-sortie (2.8), ils sont évidemment identifiable et identifiés facilement. En effet, pour un mode séquence donné $\sigma_t$, sous la Persistence Excitation (PE) conditions, la procédure d'identification peut toujours s'exécuter en empilant la relation (2.8)

pour obtenir un système d'équation solvable.

$\sigma_t$ n'est pas accesible

La procédure précédent n'est plus valide. Cependant, nous pouvons modifier la méthode proposée dans [VMS03] pour identifier le système linéaire à commutation sur $\mathbb{R}$. Cette méthode est adaptée dans notre contexte et décrite par la suite.

Chaque relation entrée-sortie (2.8) est récrite pour $t = 1, \ldots, N$ :

$$z_k^T b_t = 0 \tag{2.9}$$

avec

$z_k = [y_{k+r}, y_{k+r-1}, \cdots, y_{k+r-K}, u_k]^T \quad \in \mathbb{F}_p^{K+2}$
$b_t = [1, -a_0(\sigma_t), \ldots, -a_{K-1}(\sigma_t), -c(\sigma_t)]^T \in \mathbb{F}_p^{K+2}$

$z_k$,$b_t$ sont appelés respectivement *vecteur de régresseur* et *vecteur de paramètre* selon les séquences $\sigma_t$.

On peut définir $N$ hyperplanes $S_t$, $t = 1 \ldots, N$

$$S_t = \{z_k : z_k^T b_t = 0\}$$

L'équation suivante peut être appliquée indépendamment des séquences de commutation :

$$p_N(z_k) = \prod_{t=1}^{N} (z_k^T b_t) = \nu_N(z_k)^T h_N = 0 \tag{2.10}$$

Elle s'appelle *Hybrid Decoupling Constraint*

La multiplication est fermée dans l'anneau de polynôme $\mathbb{F}_p[z_k]$. Par conséquence, le produit $p_N(z_k)$ appartient aussi dans $\mathbb{F}_p[z_k]$.

Le vecteur $h_N \in \mathbb{F}^{M_N(K)}$ est les coefficients de *Hybrid Decoupling Constraint* et $\nu_N : z_k \in \mathbb{F}_p^{K+2} \mapsto \xi_k \in \mathbb{F}_p^{M_N(K)}$ est une *application de Véronèse* de degré $N$, les composants de $\xi_k$ correspondent à tous les monômes $M_N(K)$ (sortant dans l'ordre lexicographique à partir du produit entre les éléments de $z_k^{(i)}$ et de $z_k$). [1]

La quantité de $M_N(K)$ est donnée par :

$$M_N(K) = \begin{pmatrix} N+K+1 \\ N \end{pmatrix} = \frac{(N+K+1)!}{N!(K+1)!} \tag{2.11}$$

1. Un *ordre lexicographique* est un ordre portant sur le nom des variables et de leurs itérés, tel que :
   - $z_k^{(i)} < z_{k+l}^{(i)}$, $\forall l \in \mathbb{N}$,
   - $z_m^{(i)} < z_l^{(j)} \Rightarrow z_{m+t}^{(i)} < z_{l+t}^{(j)}$, $\forall m \in \mathbb{N}$, $\forall l \in \mathbb{N}$, $\forall t \in \mathbb{N}$,
   - $z_k^{(i)} < z_k^{(j)} \Rightarrow (z_k^{(i)})^\alpha < (z_k^{(j)})^\beta$, $\forall \alpha \in \mathbb{N}$, $\forall \beta \in \mathbb{N}$

et récrite par $M_N$ par la suite.

**Calcul de $h_N$**
Pour identifier le vecteur de paramètre $b_t$ dans (2.9), on calcule, tout d'abord, le vecteur de coefficient $h_N$ dans (2.10). On notera $\mathcal{L}_N$ - la matrice de données - qui implique $N'$ images de vecteur de régresseur $z_k$ via $\nu_N$ :

$$\mathcal{L}_N = \begin{bmatrix} \nu_N(z_{k_1}) \\ \nu_N(z_{k_2}) \\ \dots \\ \nu_N(z_{k_{N'}}) \end{bmatrix}^T \in \mathbb{F}_p^{N' \times M_N}$$

La relation suivante est vérifiée :

$$\mathcal{L}_N h_N = \mathbf{0} \tag{2.12}$$

$N'$ est un entier tel que les $\nu_N(z_{k_i})$ $(i = 1, \dots, N')$ peuvent engendrer un espace de vecteur de dimension $M_N - 1$. Cela implique :

$$rank(\mathcal{L}_N) = M_N - 1 \tag{2.13}$$

La borne minimale de $N'$ est évidemment $M_N - 1$. Si l'équation (2.13) est vérifiée, les coefficients $h_N$ peuvent être récupérés par :

$$h_N = Ker(\mathcal{L}_N) \tag{2.14}$$

**Calcul de $b_t$**
Rappelons les définitions suivantes :

**Définition 4** [Lan02] Une dérivation $D$ sur le corps $\mathbb{F}_p$ est une application $D : \mathbb{F}_p \mapsto \mathbb{F}_p$ qui est linéaire et satisfait les règles ordinaires de la dérivation, c-à-d, pour chaque élément $x, y$ dans $\mathbb{F}_p$,

$$D(x + y) = D(x) + D(y) \text{ and } D(x.y) = xD(y) + yD(x)$$

Par conséquence, la dérivation $Dp_N(z_k)$ de $p_N(z_k)$ dans (2.10) appartient aussi à l'anneau polynôme $\mathbb{F}_p[z_k]$ et vérifie :

$$Dp_N(z_k) = \frac{\partial p_N(z_k)}{\partial z_k} = \frac{\partial}{\partial z_k} \prod_{t=1}^{N} ({z_k}^T b_t) = \sum_{t=1}^{N} b_t \prod_{l \neq t}^{N} ({z_k}^T b_l) \tag{2.15}$$

On récrit (2.15) :

$$Dp_N(z_k) = b_t \prod_{l \neq t}^{N} ({z_k}^T b_l) + \sum_{i \neq t}^{N} b_i \prod_{j \neq i}^{N} ({z_k}^T b_j) \tag{2.16}$$

Considérons maintenant un vecteur arbitraire $w_t \in \mathbb{F}_p^{K+2}$, tel que, $w_t^T b_t = 0$. En remplaçant $w_t$ $(t = 1, \ldots, N)$ à l'équation (2.16), on obtient :

$$Dp_N(w_t) = b_t \prod_{l \neq t}^{N} (w_t^T b_l) = b_t.c \tag{2.17}$$

où $c$ est un scalaire. Les vecteurs de paramètre $b_t$ sont donc obéis par une normalization de (2.17).

Pour déterminer $N$ points distincts qui appartiennent aux $N$ hyperplanes $S_t$, on applique la procédure algébrique suivante :

Considérons une ligne arbitraire de direction $tv$ avec un point $w_0$ :

$$\mathcal{D} : tv + w_0 \ \forall t \in \mathcal{R}$$

La ligne $\mathcal{D}$ et tous les hyperplanes se croisent aux $N$ intersections distinctes à condition que la ligne ne soit pas en parallel avec tous les hyperplanes. Le cas de trois dimensions est illustré dans FIG. 2.1. En d'autres termes, l'équation de degré $N$

$$q_n(t) = p_N(tv + w_0) = 0 \tag{2.18}$$

a $N$ racines distinctes $\{t_i\}_{i=1}^{N}$ sous la contraint $p_N(v) \neq 0$. Par conséquence, les intersections entre cette ligne et tous les hyperplanes sont :

$$w_i = t_i v + w_0 \ \forall i \in \{1, .., N\}$$

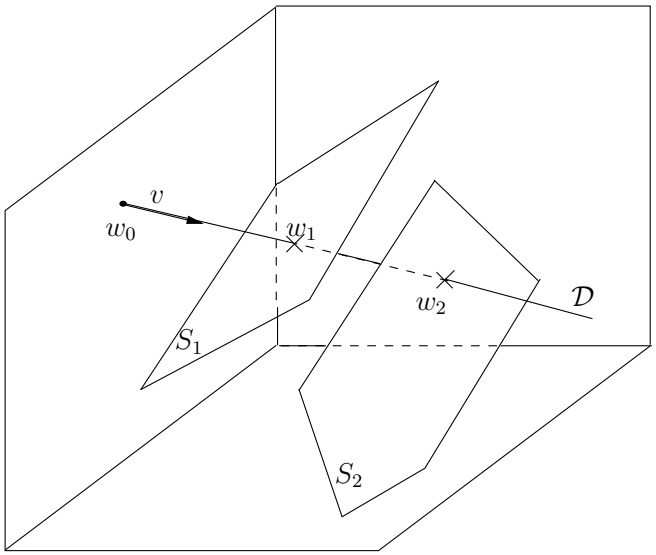


FIGURE 2.1 – Les intersections entre une ligne arbitraire $\mathcal{D}$ et deux plans $S_1$,$S_2$ dans l'espace de trois dimensions

**Remarque 1** Une autre méthode peut-être envisagée. $w_t$ appartient à un corps fini, un recherche exhaustif peut être donc effectué pour déterminer les intersections.

## 2.3 Conclusion

Ce chapitre a abordé l'invertibilité, la platitude et l'identifiabilité. Ces notions importants empruntées dans la théorie de contrôle nous donnent une base de connaissances qui utiles par la suite. Le Chapitre suivant introduit des schémas de chiffrement usuels. En particulier, nous nous intéressons aux chiffrements par flot auto-synchronisant. Une connection entre les chiffrements chaotiques et les chiffrements usuels est aussi établie.

# Chapitre 3

# Connection avec le chiffrement usuel

La sécurité des réseaux, reposant en partie sur la confidentialité des échanges d'informations, est devenue une réelle nécessité de par le développement considérable des nouvelles technologies de communication. Ainsi, la cryptographie joue un rôle prépondérant car les informations sont devenues accessibles en transitant au travers de réseaux publics.

Dans le Chapitre 2, nous avons présenté les multiples techniques de "brouillage" de l'information utilisant des systèmes dynamiques non linéaires ont été proposées depuis le début des années 90. Cependant, une attention insuffisante a été portée sur les principes de base auxquels ces techniques de "brouillage" devaient se soustraire pour garantir des niveaux de sécurité exigés en pratique. L'objectif principal de ce chapitre est d'étudier, après avoir rappelé le cadre général développé dans le Chapitre 2, la connexion entre les techniques de "brouillage" basées sur l'utilisation de dynamiques complexes et les algorithmes standards de chiffrement symétrique. Le but de cette étude est d'être en mesure de sélectionner les algorithmes pertinents de "brouillage" et de pouvoir à terme transposer aisément les techniques standard de cryptanalyse à ces algorithmes en vue de leur validation.

Ce chapitre est organisé comme suit. En Section 3.1 sont rappelés les principaux types de chiffrement standards. On s'interesse aux chiffrements particuliers dits par flot auto-synchronisant. Deux algorithmes particuliers connus sous le nom de SSS et Moustique sont ensuite détaillés et les différences structurelles sont mises en exergue. La Section 3.2 s'attache à présenter les principaux algorithmes de "brouillage", proposés depuis les années 90, basés sur l'utilisation de systèmes non linéaires exhibant des dynamiques complexes. Une comparaison avec les algorithmes de chiffrement par flot est menée. On montre comment les propriétés structurelles des systèmes dynamiques, à savoir l'inversibilité, la platitude ou l'identifiabilité paramétrique jouent un rôle fondamental à la fois pour établir une connexion ainsi que pour élaborer une nouvelle technique de synthèse de

chiffreurs symétriques autosynchrones. En Section 3.3, cette nouvelle technique de synthèse est particularisée pour la classe des systèmes dynamiques linéaires par morceaux.

## 3.1 Introduction générale à la cryptographie

### 3.1.1 Un peu d'histoire

La fonction première de la cryptographie est de cacher le sens d'un message à tous ceux qui ne sont pas autorisés à le connaître. Les quelques indications historiques qui suivent peuvent être trouvées dans de nombreuses sources, [Kah96] par exemple. Elles visent à montrer, outre l'ancienneté du problème de chiffrement, l'émergence progressive des notions de substitution, de confusion et de clé de chiffrement.

La cryptographie existe depuis que les hommes ont appris à communiquer entre eux. Vers 600 ans avant J.-C., Nabuchodonosor, roi de Babylone, écrivait le message qu'il souhaitait transmettre à ses généraux, sur le crâne préalablement rasé de ses esclaves. Il attendait que leurs cheveux repoussent avant de les envoyer chez ses généraux, qui rasaient de nouveau les cheveux des messagers pour lire le texte. Dans l'Antiquité, les Grecs employaient, en temps de guerre, un dispositif appelé une scytale. Ce dispositif consistait en une bande étroite de parchemin sur laquelle ils écrivaient après l'avoir enroulée en spirales autour d'un cylindre de bois. Une fois la bande déroulée, le texte ne pouvait être lu que par une personne possédant un cylindre de même diamètre sur lequel elle pouvait enrouler la bande.

Une méthode plus sûre qui se rapproche davantage des systèmes cryptographiques est le code de César (50 ans avant J.-C.). Pour masquer ses messages, Jules César utilisait une substitution, c'est-à-dire qu'il remplaçait chaque lettre du message par une autre lettre de l'alphabet, décalée d'une quantité fixe de la lettre d'origine. Ce code était peu sûr car l'alphabet comprenant 26 lettres, il existait seulement 26 façons différentes de chiffrer un message. Mais, la faible alphabétisation de la population le rendait assez efficace. De par sa simplicité de mise en oeuvre, il a même été réutilisé par l'armée russe pendant la première guerre mondiale. Les systèmes cryptographiques qui suivirent gardèrent ce principe de substitution.

Plus tard, en 1467, Leone Battista Alberti proposa un procédé de substitution polyalphabétique. Son principe était de remplacer chaque lettre du message par une lettre d'un autre alphabet et de changer plusieurs fois d'alphabet au cours du procédé. Vers 1500, l'abbé Jean Trithème imagina un dispositif consistant à remplacer une lettre du message par un groupe de mots. Ces groupes de mots étaient choisis de telle manière qu'un texte latin cohérent, une prière ou une glorification religieuse résultaient de la succession de ces groupes de mots.

L'inconvénient majeur de ces procédés cryptographiques fondés sur la substitution était le problème de la fréquence d'apparition des lettres. Par exemple, dans la langue française, la lettre "e" a une plus grande fréquence d'apparition dans les mots que la lettre "z". Cette fréquence d'apparition est conservée dans le texte codé, pouvant ainsi conduire à son décodage.

Pour renforcer la sécurité, les algorithmes basés sur la substitution ont été développés et améliorés. Ainsi, en 1586, le diplomate français Blaise de Vigenère proposa une technique plus élaborée, basée sur une substitution polyalphabétique. Il utilisa une clé littérale, ou mot de passe, dont chaque lettre indiquait le décalage alphabétique à appliquer sur les lettres du message. L'inconvénient de ce procédé résidait dans l'échange de la clé qui n'était pas sécurisé et qui pouvait conduire à l'interception de la clé.

En 1918, Arthur Scherbius fit breveter sa machine à crypter, appelée Enigma. Son principe fut que chaque lettre du message était remplacée par une autre, la règle de substitution changeant d'une lettre à une autre. Ce procédé permettait d'évincer le problème de la fréquence d'apparition des lettres ainsi que celui de l'échange de la clé. Cette machine fut utilisée pendant la seconde guerre mondiale. Cependant, quelque peu lente et encombrante, cette machine était inexploitable en terrain hostile. Ainsi, l'ingénieur américain Philip Johnston eut l'idée d'utiliser la langue navajo comme procédé cryptographique. La méconnaissance quasi totale de cette langue ainsi que sa construction grammaticale très particulière, la rendant impénétrable aux étrangers, décidèrent de son utilisation, lors de la campagne du Pacifique pendant la seconde guerre mondiale.

Le développement des ordinateurs, des techniques de communication et la mondialisation des échanges (Internet, commerce électronique, ...) sont confrontés à de nouveaux problèmes de sécurité de l'information. De ce fait, la cryptographie n'a plus seulement la vocation de préserver la confidentialité des données, mais elle a aussi pour rôle de préserver le contenu des messages de toute modification non souhaitée, de s'assurer de l'identité de l'émetteur et du destinataire afin d'éviter toute usurpation d'identité, ... Les systèmes cryptographiques existant jusqu'alors doivent être perfectionnés pour faire face à ces nouveaux problèmes.

Dans les années 70, Horst Feistel a mené à IBM, un projet de recherche sur le chiffrement, qui inspira plus tard le schéma de chiffrement symétrique DES. En 1976, Whitfield Diffie et Martin Hellmann proposent la cryptographie à clé publique. Ce schéma permet de pallier le problème de l'échange de clé, rencontré dans la substitution polyalphabétique de Vigenère. Le chiffrement du message est fondé sur des problèmes mathématiques difficiles à résoudre. Ce procédé est détaillé dans la Section 3.1.3, où un exemple d'algorithme à clé publique, RSA, est présenté.

Un autre procédé célèbre est la cryptographie symétrique, dont quelques exemples

sont le schéma de Vigenère et, beaucoup plus récemment, l'algorithme DES. Dans ce cas, les clés pour coder et décoder le message sont les mêmes. L'émetteur et le destinataire doivent alors s'accorder sur une clé qui doit être gardée secrète. Le chiffrement est fondé sur une combinaison complexe de substitutions. Cet algorithme est présenté dans la Section 3.1.4.

Les deux algorithmes principaux, le chiffrement à clé publique et le chiffrement symétrique, sont toujours utilisés actuellement.

### 3.1.2 Définitions

La cryptographie est l'étude de techniques mathématiques liées à la sécurité de l'information. Par sécurité de l'information, on entend confidentialité des données, intégrité des données, authentification des données et des communicants, et non répudiation des données. La confidentialité consiste à garder des données secrètes pour tous ceux qui ne sont pas autorisés à les connaître. L'intégrité des données a pour but de préserver les données de toute altération non autorisée. L'authentification des données consiste à faire le lien entre les données et leur expéditeur. L'authentification des entités consiste à s'assurer de leur identité. La non répudiation consiste à éviter que, par la suite, les communicants nient leurs actions : l'émetteur nie avoir envoyé un message et le récepteur nie avoir reçu un message.

La cryptographie consiste notamment en l'élaboration de schémas de chiffrement/déchiffrement ou *cryptosystèmes*. Le chiffrement ("encryption", en anglais) est l'opération qui consiste à transformer un message afin d'en cacher le sens à tous ceux qui ne sont pas autorisés à le connaître. Le déchiffrement ("decryption", en anglais) est l'opération inverse du chiffrement, il a pour but de récupérer l'information masquée. Un cryptosystème est l'ensemble des deux méthodes de chiffrement et de déchiffrement. En cryptographie, l'information à masquer est également appelée message ou *texte clair* ("plaintext", en anglais). Le résultat du chiffrement d'un texte clair est appelé *texte chiffré* ("ciphertext", en anglais). Le texte chiffré est le résultat d'une transformation dépendant du message et d'une clé.

Lorsqu'un cryptosystème est synthétisé, il faut s'assurer qu'il est effectivement robuste face à des attaques pirates. Cette étape de validation est appelée la cryptanalyse. Elle consiste à tester les cryptosystèmes afin de déceler leurs éventuelles faiblesses. Parmi une grande variété de mécanismes de chiffrement, les deux algorithmes principaux en cryptographie standard sont le chiffrement à clé publique et le chiffrement symétrique, présentés dans les sections suivantes.

### 3.1.3 Chiffrement à clé publique

Le chiffrement à clé publique, ou chiffrement asymétrique, a été proposé par Diffie et Hellman, en 1976. Dans un tel schéma, la clé de chiffrement est différente de celle de déchiffrement. N'importe qui peut utiliser la clé de chiffrement, ou clé publique, pour chiffrer un message, mais seul celui qui possède la clé de déchiffrement, ou clé privée, peut déchiffrer le message chiffré résultant.

De plus, la clé de déchiffrement $K^d$ ne peut pas être calculée (du moins dans un temps raisonnable) à partir de la clé de chiffrement $K^e$. Ce type de schéma repose directement sur l'existence de fonctions à sens unique. Une fonction est dite à sens unique quand il est facile de calculer $K^e$ en connaissant $K^d$, mais très difficile de calculer $K^d$ connaissant $K^e$. Parfois, des fonctions à sens unique qui possèdent en plus une trappe sont utilisées. Une fonction à sens unique est dite à trappes quand il est très difficile de calculer $K^d$ à partir de $K^e$, sauf si on connaît une information supplémentaire.

Lorsqu'Alice, l'émetteur, et Bob, le destinataire, veulent communiquer de façon sécurisée, Bob choisit une paire de clés de chiffrement et de déchiffrement $(K^e, K^d)$. Il envoie la clé publique $K^e$ à Alice, par l'intermédiaire d'un canal qui n'est pas forcément sécurisé. Alice transforme le message $m$ en texte chiffré $c = e(K^e, m)$, où $e$ représente une fonction de chiffrement, et envoie ce texte chiffré $c$ à Bob. De son côté, Bob reçoit le texte chiffré $c$ et calcule $m = d(K^d, c)$ où $d$ est une fonction de déchiffrement et $K^d$ est la clé privée connue uniquement de Bob. Ainsi, Bob récupère le message initial $m$. Ce schéma est illustré sur la FIG. 3.1.

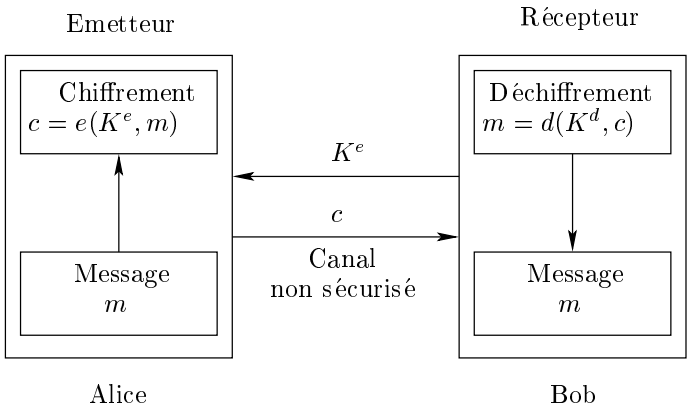


FIGURE 3.1 – Chiffrement à clé publique

Le chiffrement à clé publique ne protège pas du problème d'authentification et de non répudiation. Pour pallier le problème d'authentification, Bob envoie des "défis" à Alice qui souhaite s'identifier. Alice ne pourra résoudre ces défis que dans la mesure où elle connaît un secret, détenu par elle seule ou commun à Bob

et elle. Par exemple, pour payer par carte bancaire, le défi à relever est de taper le code secret associé à la carte, connu uniquement de son propriétaire.

Pour éviter le problème de non répudiation, un algorithme de signature numérique est ajouté. Une signature numérique est l'analogie numérique d'une signature manuscrite. La signature doit dépendre du message à signer et d'une clé secrète connue uniquement de l'entité qui signe. Une tierce entité, non corrompue, doit être capable de vérifier la signature sans avoir accès au message.

Un exemple de chiffrement à clé publique est le schéma RSA, proposé par Rivest, Shamir et Adleman, en 1978. Ce schéma est encore très largement utilisé (sites web commerciaux, par exemple). Il repose sur la difficulté de factoriser des grands nombres et s'appuie donc sur la théorie des nombres.

La génération des clés publique et privée peut être résumée par les étapes suivantes :
- Bob choisit deux grands nombres premiers (de longueur 1024 ou 2048 bits, en général), $p$ et $q$, et calcule $n = pq$.
- Bob choisit aussi, de façon aléatoire, un entier $K^e < n$ qui est premier avec $(p-1)(q-1)$.
- Il calcule $K^d$ tel que $K^e K^d = 1 \bmod (p-1)(q-1)$.
- Bob publie la clé publique, formée de $(K^e, n)$, et les entiers $p$ et $q$ sont détruits pour ne pas être divulgués. Il garde la clé privée $K^d$.

Pour envoyer un message à Bob, Alice calcule $c = m^{K^e} \bmod n$ et envoie $c$ à Bob.
Pour déchiffrer $c$, Bob calcule $m = c^{K^d} \bmod n$.
Dans ce schéma, la fonction de chiffrement est une fonction à sens unique et possède une trappe. En effet, toute personne connaissant la clé publique $(K^e, n)$ et la factorisation de $n$ peut calculer $K^d$.

### 3.1.4 Chiffrement symétrique

Par opposition au chiffrement à clé publique, le chiffrement symétrique est aussi appelé chiffrement à clé secrète. La clé de chiffrement peut être calculée à partir de la clé de déchiffrement et vice versa. En général, les clés de chiffrement et de déchiffrement sont identiques. L'émetteur et le destinataire doivent se mettre d'accord préalablement sur une clé qui doit être gardée secrète, car la sécurité d'un tel algorithme repose sur cette clé.

Le chiffrement à clé publique et le chiffrement symétrique présentent chacun des avantages [MOV96]. Par exemple, le temps de chiffrement/déchiffrement du chiffrement à clé publique est supérieur à celui du chiffrement symétrique. Un des problèmes principaux du chiffrement symétrique est l'échange préalable de la clé secrète. Le chiffrement à clé publique peut être préféré pour générer de petites séquences comme des signatures ou des clés secrètes pour le chiffrement symétrique. Le chiffrement symétrique peut être préféré pour chiffrer des grandes quantités

de données.

Les schémas de chiffrement symétrique peuvent être classés en deux catégories, le chiffrement par blocs et le chiffrement par flot, détaillés ci-après.

**Chiffrement par blocs**

Dans un schéma de chiffrement par blocs, le message est divisé en blocs de bits, de longueur fixe. Chaque bloc est chiffré l'un après l'autre. Le chiffrement peut être effectué par substitutions (les bits d'un bloc sont substitués par d'autres bits) et par transpositions (les bits d'un bloc sont permutés entre eux). La substitution permet d'ajouter de la *confusion*, c'est-à-dire de rendre la relation entre le message et le texte chiffré aussi complexe que possible. La transposition permet d'ajouter de la *diffusion*, c'est-à-dire de réarranger les bits du message afin d'éviter que toute redondance dans le message ne se retrouve dans le texte chiffré.

On distingue le chiffrement par blocs itératifs. Une fonction constituée de combinaisons complexes de substitutions et de transpositions, appelée fonction de tour ou fonction de ronde, est appliquée itérativement. Une itération est appelée un tour ou une ronde. Chaque ronde prend en entrée la sortie de la ronde précédente (ou un bloc du texte clair pour la première ronde) et chiffre cette entrée à l'aide de la fonction de ronde et d'une sous-clé de ronde générée à partir de la clé secrète $K$. La fonction de chiffrement n'est pas la fonction de ronde, mais elle est constituée par l'ensemble de toutes les rondes.

Un exemple de chiffrement par blocs itératifs est le célèbre schéma DES (**D**ata **E**ncryption **S**tandard), adopté par le gouvernement américain, en 1977, comme algorithme de chiffrement standard officiel. Dans ce schéma, le texte clair est divisé en blocs de longueur 64 bits. La clé a également une longueur de 64 bits, dont 56 bits sont générés aléatoirement et utilisés dans l'algorithme et dont 8 bits sont utilisés pour la détection d'erreurs lors de la transmission. Le chiffrement d'un bloc s'effectue avec 16 rondes. La clé secrète $K$ est dérivée de 16 "sous-clés" $K_i$, une pour chaque ronde. Chaque entrée de ronde est partagée en une partie gauche $L_i$ et une partie droite $R_i$, de même longueur. Pour $i = 0, \ldots, 15$, les quantités $R_{i+1}$ et $L_{i+1}$ sont calculées :

$$\begin{cases} R_{i+1} = L_i \oplus f(R_i, K_i) \\ L_{i+1} = R_i \end{cases} \tag{3.1}$$

où $f$ est la fonction de ronde.

Pour renforcer la sécurité, il existe des variantes du DES qui consistent à utiliser une clé $K$ de longueur plus importante et à répéter plusieurs fois l'algorithme sur chaque bloc, comme le triple-DES.

Les longueurs des clés ne permettent pas toujours de résister à des attaques de plus en plus performantes grâce au progrès des ordinateurs. Pour pallier ce problème, le schéma DES est amélioré et devient le schéma AES (**A**dvanced **E**ncryption **S**tandard), en 1997.

### 3.1.5 Chiffreurs symétriques usuels par flot

On distingue les parties émetteur (ou chiffreur) et récepteur (ou déchiffreur) avec leur générateur de clé respectif. $m$ désigne le message clair et $c$ le message chiffré. Côté émetteur, une entité caractérisée par une fonction $e$ dépendant à la fois du message clair $m$ et de la clé $k^e$ assure le chiffrement. Côté récepteur, le déchiffrement est assuré par une fonction $d$ dépendant de la clé $k^d$ et du message chiffré $c$. Le message chiffré $c$ est véhiculé au travers d'un canal supposé public et peut donc être intercepté par une tierce partie appelée pirate. Dans un procédé de chiffrement symétrique, le cadre de notre étude, les clés des parties émetteur et récepteur doivent être identiques, c'est-à-dire $k^e = k^d$, pour retrouver correctement l'information et assurer $\hat{m} = m$.

Pour le chiffrement symétrique par flot, le message clair (*resp.* message chiffré) est constitué d'une séquence de symboles clairs $m_k$ (*resp.* symboles chiffrés $c_k$). Notons que la terminologie message clair (*resp.* message chiffré) lorsqu'on se réfère au symbole $m_k$ (*resp.* symbole $c_k$) est souvent abusivement conservée. La clé $k^e$ du chiffreur (*resp.* $k^d$ du déchiffreur) est remplacée par une séquence (ou flot) de clés notées $z_k$ pour le chiffrement (*resp.* $\hat{z}_k$ pour le déchiffrement). Conformément au principe du chiffrement symétrique, les quantités $z_k$ et $\hat{z}_k$, appelées clés courantes, doivent être égales à chaque instant pour retrouver correctement le message clair et donc assurer $\hat{m}_k = m_k$ pour tout $k$. En d'autres termes les séquences de clés courantes doivent être synchronisées. Les générateurs côté émetteur et récepteur produisant les séquences de clés courante dépendent d'un paramètre $\theta$ qui constitue la clé secrète. Deux types de chiffrement par flot se distinguent par leur générateurs de clés : synchrones et autosynchrones.

#### 3.1.5.1 Chiffreurs par flot synchrones

Un chiffreur par flot synchrone obéit aux équations (3.2)

$$\begin{cases} z_k = \sigma^s(z_{k-1}) \\ c_k = e(z_k, m_k) \end{cases} \tag{3.2}$$

La clé courante $z_k$ est générée par un système dynamique autonome caractérisé par une fonction $\sigma^s$. Afin de pouvoir récupérer correctement le message clair $m_k$, les générateurs de clé côté émetteur et récepteur doivent être parfaitement synchronisés. Etant donné qu'il n'y a aucun couplage entre eux, la seule possibilité pour garantir leur synchronisation est de les initialiser à la même valeur $z_0 = \hat{z}_0$. Ainsi, la condition initiale des générateurs $z_0 = \hat{z}_0$ constitue la clé secrète $\theta$.

#### 3.1.5.2 Chiffreurs par flot autosynchrones

Un chiffreur par flot autosynchrone obéit aux équations (3.3)

$$\begin{cases} z_k = \sigma_\theta^{ss}(c_{k-l}, \ldots, c_{k-l-M}) \\ c_k = e(z_k, m_k) \end{cases} \tag{3.3}$$

Dans ce cas, le générateur de clé n'est pas un système dynamique autonome. Il est caractérisé par une fonction statique $\sigma_\theta^{ss}$ dépendant d'un nombre fini de textes chiffrés précédents $c_{k-i}$ et paramétrée par $\theta$ qui joue le rôle de clé secrète. La propriété essentielle de ce type de chiffrement est la capacité d'auto synchronisation des générateurs de clé. En effet, après un nombre fini d'itérations correspondant au nombre de chiffrés $c_{k-i}$ dont dépend $\sigma_\theta^{ss}$, les séquences des clés $z_k$ et $\hat{z}_k$ sont identiques puisque $\sigma_\theta^{ss}$ dépend exactement des mêmes arguments. Ce schéma de chiffrement est intéressant pour ses propriétés d'auto-synchronisation. En effet, aucun protocole ou trame de resynchronisation n'est nécessaire en cas de désynchronisation impromptue. On y a recours dans les situations où on doit faire face à de fortes contraintes en terme de débit.

Les équations (3.3) ne sont en réalité que des formes canoniques, en particulier la fonction $\sigma_\theta^{ss}$. Le générateur de clé peut être en fait un système dynamique mais aux propriétés telles qu'une réécriture via une fonction statique $\sigma_\theta^{ss}$ est possible comme cela est illustré au travers des deux exemples qui suivent.

### 3.1.6 Exemples de chiffreurs autosynchronisants

#### 3.1.6.1 SSS

L'algorithme SSS a été proposé dans [HPRM04]. L'architecture générale est représentée sur le schéma-bloc de la FIG. 3.2.

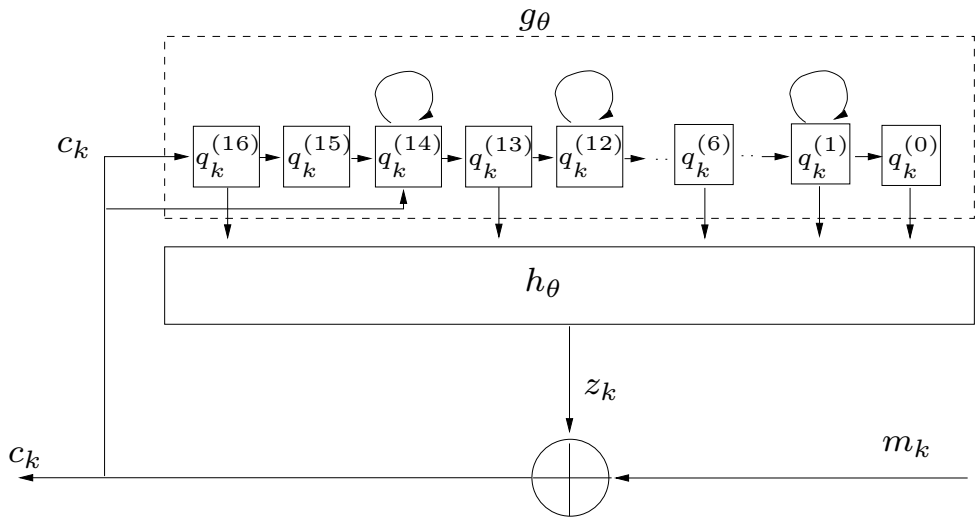


FIGURE 3.2 – Le schéma bloc de SSS côté émetteur

Les opérations suivantes seront utilisées pour décrire l'algorithme.
- $x >>> n$ est une rotation de $n$ bits à droite du mot $x$
- $S_\theta(x) = SBOX_\theta(x_H) \oplus x$ avec $x_H$ l'octet de poids fort de $x$ est le OU exclusif entre $x$ et le résultat de la fonction $SBOX_\theta$, combinaison de deux tables de substitution appelées Skipjack S-box et Q-box [HPRM04] paramétrées par la clé secrète $\theta$

Le générateur de clé est un système dynamique caractérisé par une fonction de transition $g_\theta$ et de sortie $h_\theta$.

$$\begin{cases} q_{k+1} &= g_\theta(q_k, c_k) \\ z_k &= h_\theta(q_k) \end{cases} \tag{3.4}$$

Le vecteur d'état $q_k$ est de dimension $n = 17$ correspondant au nombre de registres à décalage (d'une longueur de 16 bits) auxquels sont associées chacune des composantes $q_k^{(j)}$. Les dynamiques notées $g_\theta^j$ des $q_k^{(j)}$ sont indépendantes et décrites par :

$$\begin{array}{rcl} q_{k+1}^{(16)} &=& c_k \\ q_{k+1}^{(j)} &=& q_k^{(j+1)} \ (j = 0, 2, \ldots, 11, 13, 15) \\ q_{k+1}^{(14)} &=& q_k^{(15)} + S_\theta(c_k >>> 8) \\ q_{k+1}^{(12)} &=& S_\theta(q_k^{(13)}) \\ q_{k+1}^{(1)} &=& q_k^{(2)} >>> 8 \end{array} \tag{3.5}$$

L'état initial du registre 16 vérifiant $q_1^{(16)} = c_0$, il est facile de voir qu'au bout de 16 itérations, l'état interne $q_k$ ne dépend plus que des 16 valeurs précédentes du message chiffré $c_k$. Il existe donc pour tout $k \geq 16$ une fonction $l_\theta$ vérifiant

$$q_k = l_\theta(c_{k-16}, \ldots, c_{k-1}) \tag{3.6}$$

La fonction de sortie $h_\theta$ délivrant le flot de clés courantes $z_k$ est définie par :

$$z_k = h_\theta(q_k) = A_\theta >>> 8 \oplus q_k^{(0)} \tag{3.7}$$

avec $A_\theta = S_\theta\Big(S_\theta(q_k^{(0)} + q_k^{(16)}) + q_k^{(1)} + q_k^{(6)} + q_k^{(13)}\Big)$

Finalement, en combinant l'équation (3.7) et (3.6), le générateur de clé peut être réécrit sous la forme canonique (3.3) :

$$\begin{array}{rl} z_k & = h_\theta(l_\theta(c_{k-16}, .., c_{k-1})) \\ & = \sigma_\theta^{ss}(c_{k-16}, .., c_{k-1}) \end{array} \tag{3.8}$$

La fonction de chiffrement $e$ est un ou exclusif entre le message clair $m_k$ et la clé courante $z_k$ :

$$c_k = z_k \oplus m_k \tag{3.9}$$

La fonction de déchiffrement $d$ est donc également un ou exclusif entre le message chiffré $c_k$ et la clé courante $\hat{z}_k$

$$\hat{m}_k = c_k \oplus \hat{z}_k = m_k \text{ si } \hat{z}_k = z_k \tag{3.10}$$

#### 3.1.6.2 Moustique

Le chiffreur autosynchrone Moustique [DP06] est la dernière version de l'algorithme original KNOT [DGV92]. C'est un algorithme de chiffrement binaire. Son principe est dérivé de la méthode de synthèse proposé pour la première fois par [Mau91] qui consistait à remplacer les registres à décalage des générateurs de clés par une association (série, parallèle) d'automates à mémoire finie.
Pour Moustique, le générateur de clé est un système dynamique caractérisé par une fonction de transition $g_\theta$ et fonction de sortie $h$.

$$\begin{cases} q_{k+1} &= g_\theta(q_k, c_k) \\ z_k &= h(q_k) \end{cases} \tag{3.11}$$

Le vecteur d'état $q_k$ est de dimension $n = 96$. Moustique se différencie de SSS à la fois au niveau de la fonction de transition et de la fonction de sortie.
Concernant la fonction de transition $g_\theta$, chaque composante $q_k^{(j)} \in \{0, 1\}$ obéit à une dynamique $g_\theta^j$ de la forme :

$$q_{k+1}^{(j)} = g_\theta^j(q_k^{(j-1)}, q_k^{(j-2)}, ..., q_k^{(1)}, c_k) \; j = 1, \ldots, n \tag{3.12}$$

Ainsi, la $j^{ème}$ composante de $q_{k+1}$ ne dépend plus d'une seule composante de $q_k$ (comme c'est le cas avec les registres à décalage dans SSS), mais dépend de plusieurs composantes de $q_k$, plus précisément des composantes $q_k^{(l)}$ avec $l < j$. Ainsi, la fonction $g_\theta$ est « triangulaire » ; cela garantit que $q_k$ ne dépend plus de la condition initiale $q_0$ après $n$ itérations. De même que pour SSS, il existe donc une fonction $l_\theta$ qui permet de réécrire (3.12) sous une forme strictement équivalente pour $k \geq n$ et ne dépend plus que des messages chiffrés $c_{k-i}$ précédents

$$q_k = l_\theta(c_{k-n}, ..., c_{k-1}) \tag{3.13}$$

A la différence de SSS, la fonction de sortie de Moustique n'est pas une simple fonction statique. Elle est en réalité implémentée sous la forme d'un « pipeline » (Fig. 3.3). Le calcul de la clé courante s'opère de façon séquentielle en impliquant des étages successifs. Le « pipeline » de Moustique comprend $b_s = 9$ étages. Chaque étage correspond à une fonction spécifique $s_i$ $(i = 0, \ldots, b_s - 1)$ dont l'argument d'entrée est le résultat de la sortie de l'étage précédent. L'argument d'entrée de la fonction $s_0$ est $q_k$ et on a $s_0(q_k) = q_k$. Notons qu'aucune fonction $s_i$ ne dépend de la clé secrète $\theta$. La fonction de sortie $h$ (qui n'est donc pas, à la différence de SSS, paramétére par $\theta$) s'écrit comme une composition de $b_s$ fonctions et la clé courante calculée à partir d'un état $q_k$ n'est disponible qu'à l'instant $k + b_s$ :

$$z_{k+b_s} = s_8(s_7(...(s_0(q_k)))) = h(q_k) \tag{3.14}$$

En combinant (3.13) et (3.14), on obtient la fonction $\sigma_\theta^{ss}$

$$\begin{aligned} z_{k+b_s} &= h(l_\theta(c_{k-n}, ..., c_{k-1})) \\ &= \sigma_\theta^{ss}(c_{k-n}, ..., c_{k-1}) \end{aligned} \tag{3.15}$$

On comprend alors l'intérêt matériel du pipeline : une augmentation de la complexité de la fonction de sortie mais qui nécessite un seul cycle d'horloge pour générer la clé dynamique, chaque fonction $s_i$ étant calculée en parallèle. Le prix à payer est un délai $b_s$ entre le message clair et le message chiffré correspondant. La question du temps de calcul est abordée dans [DK05][DP06].

La fonction de chiffrement $e$ est un ou exclusif entre le message clair $m_k$ et la clé courante $z_{k+b_s}$ :

$$c_{k+b_s} = z_{k+b_s} \oplus m_k \tag{3.16}$$

La fonction de déchiffrement $d$ est donc également un ou exclusif entre le message chiffré $c_{k+b_s}$ et la clé courante $\hat{z}_k$ et vérifie

$$\hat{m}_{k+b_s} = c_{k+b_s} \oplus \hat{z}_k = m_k \text{ si } \hat{z}_k = z_{k+b_s} \tag{3.17}$$

## 3.2 Procédés de "brouillage" et comparaison structurelle

On appelle procédés de "brouillage" les algorithmes qui ont été proposés dans la littérature depuis les années 90 basés sur l'utilisation des dynamiques complexes dans le contexte de la sécurité des communications. Un description exhaustive est donnée dans [MAD08]. On ne retient ici que les deux principaux pour notre étude comparative.

### 3.2.1 Masquage additif

Le masquage additif a été le premier schéma de "brouillage" proposé en 1993 [WO93][MVH93b]. Ses équations de fonctionnement et le schéma-bloc associé sont donnés respectivement par (1.3) et la FIG. 1.1.

$$\begin{cases} x_{k+1} &= f_\theta(x_k) \\ y_k &= h_\theta(x_k) + m_k \end{cases} \tag{3.18}$$

Le système dynamique exhibant une dynamique complexe est caractérisé par son vecteur d'état $x_k \in \mathbb{R}^n$, sa nonlinéarité $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ et sa fonction de sortie $h : \mathbb{R}^n \mapsto \mathbb{R}$. Les fonctions $f$ et $h$ sont paramétrées par $\theta$ qui constitue la clé secrète.

Le principe du "brouillage" est basé sur l'addition entre le signal de sortie $h_\theta(x_k)$ du système dynamique émetteur et le message clair $m_k$. Le résultat noté $y_k$ constitue le message chiffré. La récupération du message clair nécessite un système dynamique identique côté récepteur capable de se synchroniser parfaitement avec le système émetteur afin que les deux vecteurs d'état respectifs $x_k$ et $\hat{x}_k$ soient égaux.

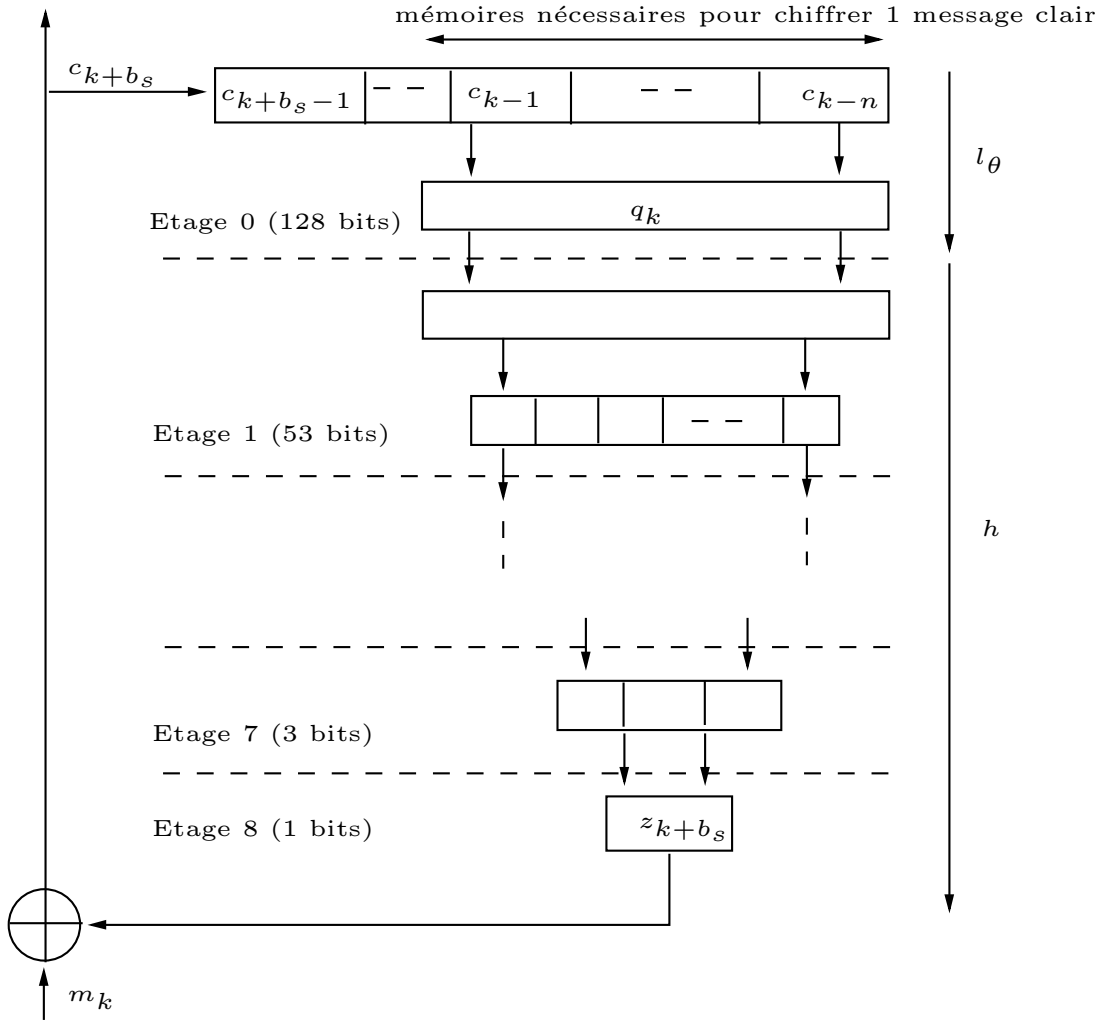


FIGURE 3.3 – Le schéma bloc de Moustique côté récepteur

Le message clair est alors reconstitué par opération de soustraction $\hat{m}_k = y_k - \hat{y}_k$.

Dans le schéma proposé par [WO93][MVH93b], une synchronisation parfaite des séquences $x_k$ et $\hat{x}_k$ est attendue sans que les systèmes dynamiques soient initialisés à la même valeur. Cependant l'ajout de $m_k$ au niveau de la sortie côté émetteur constitue une perturbation agissant sur le canal de transmission et celle-ci ne peut en général pas être totalement rejetée. La synchronisation des vecteurs d'état qui jouent en réalité le rôle de clé courante ne peut donc pas être parfaitement garantie. Ce schéma apparaît comme une version dégradée du chiffrement par flot synchrone et ne présente donc aucun intérêt.

### 3.2.2 Inclusion

La technique par inclusion obéit à la équation (3.19) et le schéma-bloc associé est représenté sur la FIG. 1.4.

$$\begin{cases} x_{k+1} = f_\theta(x_k, m_k) \\ y_k = h_\theta(x_k, [m_k]) \end{cases} \tag{3.19}$$

Cette technique consiste à injecter le message clair $m_k$ dans la dynamique du système. La sortie $y_k$ du système dynamique constitue le message chiffré. Notons que $m_k$ peut également être injecté au niveau de la sortie. Cette éventualité est caractérisée par la présence des crochets [.] dans l'équation de sortie de (3.19). La récupération du message clair $m_k$ nécessite une inversion dynamique $g$ côté récepteur. Cette inversion, dans la théorie du contrôle, est appelée "inversion à gauche". De la même manière qu'un lien a été établi entre le masquage additif et le chiffrement par flot synchrone, la recherche d'un lien éventuel entre la technique par inclusion et le chiffrement par flot s'impose.

### 3.2.3 Une nouvelle approche pour la synthèse de chiffreurs autosynchrones

La Proposition suivante est le résultat central de notre étude.

**Proposition 2** *Le chiffrement par inclusion (3.19) est équivalent à un chiffrement par flot autosynchrone si le système dynamique (3.19) est inversible à gauche, plat et possède un degré relatif fini.*

**Preuve 2** : si (3.19) est plat, alors (2.7) est vérifié. S'il possède un degré relatif fini, alors (2.2) est également vérifié. Enfin, s'il est inversible à gauche, pour tout vecteur d'état $x_k$, la fonction $h^{(r)}$ est inversible et $m_k$ se déduit donc de manière unique à partir de $y_{k+r}$. En identifiant (2.7) et (3.25) avec les équations (3.3), on en déduit que le chiffrement par inclusion (3.19) est équivalent au chiffrement par flot autosynchrone avec :
- un générateur de clé $\mathcal{F}$
- une clé courante $x_k$

- un message chiffré $y_{k+r}$ correspondant au message clair $m_k$ (le délai correspondant à un pipeline virtuel à $r$ étages)
- une fonction de chiffrement $h^{(r)}$

La preuve de ce théorème est constructive car elle révèle une nouvelle technique de synthèse de chiffreurs autosynchrones inédite à ce jour. Le recours à des systèmes dynamiques plats constitue en effet une alternative aux techniques existantes : registres à décalage (comme pour SSS) ou fonctions "triangulaires" (comme pour Moustique).

## 3.3 Exemple

Le but de cette section est d'illustrer comment se traduit, pour la classe des systèmes linéaires par morceaux, la démarche générale de conception d'un algorithme de chiffrement par flot autosynchrone décrite au travers de la preuve de la Proposition 2. En particulier, on montre comment tester l'inversibilité à gauche et la platitude. Ces systèmes obéissent à :

$$\left\{ \begin{array}{lcl} x_{k+1} & = & A_{\sigma(k)} x_k + B_{\sigma(k)} m_k \\ y_k & = & C_{\sigma(k)} x_k + D_{\sigma(k)} m_k \end{array} \right. \tag{3.20}$$

où $x_k \in \mathbb{R}^n$, $m_k \in \mathbb{R}$ et $y_k \in \mathbb{R}$. Toutes les matrices $A_{\sigma(k)} \in \mathbb{R}^{n\times n}$, $B_{\sigma(k)} \in \mathbb{R}^{n\times 1}$, $C_{\sigma(k)} \in \mathbb{R}^{1\times n}$ et $D_{\sigma(k)} \in \mathbb{R}$ appartiennent à des ensembles finis $(A_j)_{1\leq j\leq J}$, $(B_j)_{1\leq j\leq J}$, $(C_j)_{1\leq j\leq J}$ et $(D_j)_{1\leq j\leq J}$. A chaque instant $k$, l'index $j$ correspond au "mode" du système qui résulte d'une fonction de commutation $\sigma : k \in \mathbb{N} \mapsto j = \sigma(k) \in \{1, \ldots, J\}$.

On considère l'exemple suivant :

$$\begin{array}{ll} A_{\sigma(k)} = \begin{pmatrix} q^1_{\sigma(k)} & 1 \\ 0.5 & 0 \end{pmatrix} & B_{\sigma(k)} = \begin{pmatrix} 0 \\ q^2_{\sigma(k)} \end{pmatrix} \\ C_{\sigma(k)} = (1\ 0) & D_{\sigma(k)} = 0 \end{array} \tag{3.21}$$

**Degré relatif et inversibilité à gauche**

On cherche une condition algébrique d'inversibilité à gauche pour (3.20). Dans cette perspective, écrivons l'expression de la sortie $y_{k+i}$ en itérant (3.20)

$$y_{k+i} = C_{\sigma(k+i)} A^{\sigma(k+i-1)}_{\sigma(k)} x_k + \sum_{j=0}^{j=i} \mathcal{T}^{i,j}_{\sigma(k)} m_{k+j} \tag{3.22}$$

avec

$$\begin{array}{l} \mathcal{T}^{i,j}_{\sigma(k)} = C_{\sigma(k+i)} A^{\sigma(k+i-1)}_{\sigma(k+j+1)} B_{\sigma(k+j)} \text{ si } j \leq i-1 \\ \mathcal{T}^{i,i}_{\sigma(k)} = D_{\sigma(k+i)} \end{array} \tag{3.23}$$

et avec la matrice de transition définie par :

$$\begin{array}{rcl} A^{\sigma(k_1)}_{\sigma(k_0)} & = & A_{\sigma(k_1)} A_{\sigma(k_1-1)} \ldots A_{\sigma(k_0)} \text{ si } \ k_1 \geq k_0 \\ & = & \mathbf{1}_n \ \text{ si } \ k_1 < k_0 \end{array}$$

$\mathbf{1}_n$ étant la matrice identité de dimension $n$.
Il en découle directement le résultat suivant. Le degré relatif $r$ de (3.20) est
– $r = 0$ si $\mathcal{T}^{0,0}_{\sigma(k)} \neq 0$ pour tout $k$
– le plus petit entier $r < \infty$ tel que pour tout $k$

$$\begin{array}{l} \mathcal{T}^{i,j}_{\sigma(k)} = 0 \text{ pour } \; i = 0, \ldots, r-1 \;\text{ et }\; j = 0, \ldots, i \\ \mathcal{T}^{r,0}_{\sigma(k)} \neq 0 \end{array} \tag{3.24}$$

Si (3.20) a un degré relatif $r$, sa sortie au temps $k + r$ s'écrit :

$$y_{k+r} = C_{\sigma(k+r)} A^{\sigma(k+r-1)}_{\sigma(k)} x_k + \mathcal{T}^{r,0}_{\sigma(k)} m_k \tag{3.25}$$

et il est inversible à gauche puisque pour tout $x_k$, on peut retrouver $m_k$ de manière unique à partir de $y_{k+r}$, la matrice $\mathcal{T}^{r,0}_{\sigma(k)}$ étant toujours inversible, (3.24) étant vérifié.
*Application*
Le degré relatif de (3.20) pour les matrices particulières (3.21) est $r = 2$ puisque $\mathcal{T}^{i,j}_{\sigma(k)} = 0$ pour $\; i = 0, 1 \;$ et $\; j = 0, \ldots, i$, $\mathcal{T}^{2,0}_{\sigma(k)} = C_{\sigma(k+2)} A_{\sigma(k+1)} B_{\sigma(k)} \neq 0$ pour tout $k$. Il vient :

$$y_{k+2} = q^1_{\sigma(k+1)} y_{k+1} + 0.5 y_k + q^2_{\sigma(k)} m_k \tag{3.26}$$

**Platitude**
Pour tester si le système est plat, on cherche à exprimer chaque composante $x^{(i)}_k$ $(i = 1, 2)$ de l'état $x_k$ en fonction uniquement de la sortie et éventuellement de ses itérés. La méthode proposée est basée sur une technique d'élimination. Il existe de nombreux algorithmes d'élimination (cf. [Wan91] pour une étude comparative), notamment ceux dérivés de la théorie des résultants, des ensembles caractéristiques ou des bases de Gröbner. Mis à part certains cas particuliers, il est impossible d'affirmer qu'une méthode et meilleure qu'une autre. Nous avons choisi ici d'utiliser le logiciel libre Maxima (disponible à l'adresse http ://maxima.sourceforge.net) qui est basé sur la théorie des résultants [Wan91].
Trois étapes sont nécessaires pout tester la platitude :
– définir les équations d'état
– écrire les itérés successifs des équations d'état
– spécifier les variables à éliminer
Pour la composante $x^{(1)}_k$, la réponse est évidente et ne nécessite aucun calcul car on a :

$$x^{(1)}_k = y_k \tag{3.27}$$

Afin de comprendre le script Maxima correspondant à la recherche de l'expression de $x^{(2)}_k$, rappelons quelques notations utilisées. Toute variable $x^{(i)}_k$ (*resp.* $y_k$ et $m_k$) est notée `xik` (*resp.* `yk` et `mk`), les itérés de rang $l$ sont notés `xikl` (*resp.* `ykl` et `mkl`). Une équation doit toujours être labellée. Le label est de la forme `eqj` où `j` correspond à la $j^{ieme}$ équation. Les composantes non constantes des matrices $A_{\sigma(k)}$ et $B_{\sigma(k)}$, à savoir $q^1_\sigma(k)$ et $q^2_{\sigma(k)}$ sont notées `q1k` et `q2k` ; les itérés de rang $l$ à savoir $q^1_{\sigma(k+l)}$ et $q^2_{\sigma(k+l)}$ notés `q1kl` et `q2kl`. Enfin, une équation est

toujours comprise comme une relation égalant zéro. Ces quelques commentaires permettent de comprendre les scripts ci-dessous.
On spécifie tout d'abord les équations du système :

```
(%i1) eq1:x1k1-q1k*x1k-x2k;
(%i2) eq2:x2k2-0.5*x1k-q2k*mk;
(%i3) eq3:yk-x1k;
```

On écrit alors les équations d'état itérées.

```
(%i4) eq4:x1k2-q1k1*x1k1-x2k1
(%i5) eq5:x2k2-0.5*x1k1-q2k1*mk1;
(%i6) eq6:yk1-x1k1;
(%i7) eq7:y1k2-x1k2;
```

Pour obtenir l'expression (si elle existe) de la composante $x_k^{(2)}$ de $x_k$ en fonction uniquement de la sortie et éventuellement de ses itérés, on fait appel à la fonction `eliminate` de Maxima qui utilise la technique des résultants. On spécifie la liste des équations à utiliser et des variables à éliminer à savoir toutes les variables sauf `x1k` (qui n'est rien d'autre que `yk`) et `x2k`.

```
eliminate([eq1,eq2,eq3,eq4,eq5,eq6,eq7],
[mk,mk1,x1k1,x1k2,x2k1,x2k2]);
```

On obtient

$$x_k^{(2)} = y_{k+1} - q^1_{\sigma(k)} y_k \tag{3.28}$$

D'après (3.27) et (3.28), les deux composantes $x_k^{(i)}$ $(i = 1, 2)$ du vecteur d'état ne dépendent pas de l'entrée mais uniquement de la sortie et de ses itérés donc le système est plat.
Conclusion : le système (3.20) possède un degré relatif fini $r = 2$, il est inversible à gauche et plat. Conformément à la Proposition 1, (3.20) est équivalent à un chiffreur autosynchrone avec :
- un générateur de clé où $\mathcal{F}$ est donnée par les équations (3.27) et (3.28)
- une clé courante $x_k$
- un message chiffré $y_{k+2}$ correspondant au message clair $m_k$ (le délai correspondant à un pipeline virtuel à 2 étages)
- une fonction de chiffrement $h^{(2)}$ donnée par (3.26)

## 3.4 Conclusion

Dans ce chapitre une étude comparative entre les techniques de "brouillage" de l'information basées sur l'utilisation de dynamiques complexes et les algorithmes standards de chiffrement symétrique a été menée. On a montré et illustré comment les propriétés structurelles des systèmes dynamiques, à savoir l'inversibilité, la platitude jouent un rôle fondamental pour la synthèse de chiffreurs symétriques

autosynchrones.

Une issue majeure pour évaluer la sécurité d'un algorithme de chiffrement est la complexité de recouvrement de la clé secrète. Tout algorithme de chiffrement doit pouvoir faire face au moins à l'attaque la plus basique, à savoir l'attaque gloutonne (ou recherche exhaustive). Cette attaque consiste à balayer toutes les valeurs possibles de clés appartenant à un ensemble de cardinalité fini (espace des clés). Le pirate est supposé disposer d'un grand nombre de paires message clair - message chiffré correspondant. Cette hypothèse est vérifiée dans une attaque à texte clair choisi. Afin que l'espace des clés ne puisse être réduit par cette approche, il ne faut pas que pour une même paire de messages clair - chiffré, plusieurs valeurs de clés conviennent. Si tel est le cas, des valeurs de clés sont redondantes et le cryptosystème peut être plus rapidement cassé par attaque gloutonne. Or, l'unicité de la clé correspond à la propriété d'identifiablité des paramètres $\theta$ du système dynamique.

# Bibliographie


[BRV05] P. Barthélemy, R. Rolland, and P. Véron, editors. *Cryptographie*. Hermès Science, 2005.

[CH00] Cruz C. and Nijmeijer H. Synchronization through filtering. *International Journal of Bifurcation and Chaos*, 110(4) :763–775, 2000.

[DGV92] J. Daemen, R. Govaerts, and J. Vandewalle. On the design of high speed self-synchronizing stream ciphers. In *Proc. of the ICCS/ISITA'92 conference*, volume 1, pages 279–283, Singapore, November 1992.

[DK05] J. Daemen and Paris K. The self synchronizing stream cipher mosquito : estream documentation, version 2. Technical report, eStream Project, 2005. Available at :www.ecrypt.eu.org/stream/p3ciphers/mosquito/mosquito.pdf.

[DKVS98] F. Dachselt, K. Kelber, J. Vandewalle, and W. Schwarz. Chaotic versus classical stream ciphers – a comparative study. In *Proc. of Int. Symp. on Circuits and Systems ISCAS'98*, volume IV, pages 518–521, Monterey, June 1998.

[DP06] J. Daemen and K. Paris. The self synchronzing stream cipher moustique. Technical report, eStream Project, 2006. Available at : http ://www.ecrypt.eu.org/stream/p3ciphers/mosquito/mosquito_p3.pdf.

[FLMR95] M. Fliess, J. Levine, P. Martin, and P. Rouchon. Flatness and defect of non-linear systems : introductory theory and examples. *Int. Jour. of Control*, 61(6) :1327–1361, 1995.

[FM01] M. Fliess and R. Marquez. Une approche intrinsèque de la commande prédictive linéaire discrète. *Journal Européen des Systèmes Automatisés*, 35 :127–147, 2001.

[GKS97] M. Gotz, K. Kelber, and W. Schwarz. Discrete-time chaotic encryption systems - part 1 : statistical design approach. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 44(10) :963–970, Oct. 1997.

[Has98] M. Hasler. Synchronization of chaotic systems and transmission of information. *International Journal of Bifurcation and Chaos*, 8(4), April 1998.

[HPM93] Dedieu H., Kennedy M. P., and Hasler M. Chaos shift keying : modulation and demodulation of a chaotic carrier using self-synchronizing

chua's circuits. *IEEE Trans. Circuits. Syst. II : Anal. Digit. Sign. Process*, 40 :634–642, 1993.

[HPRM04] P. Hawkes, M. Paddon, G. G. Rose, and W. V. Miriam. Primitive specification for sss. Technical report, e-Stream Project, 2004. Available at : http ://www.ecrypt.eu.org/stream/ciphers/sss/sss.pdf.

[Isi95] A. Isidori. *Nonlinear control systems*. Communications and control engineering series. Springer, 1995.

[Kah96] D. Kahn. *The codebreakers*. Scribner Book Company, 1996.

[Koc01] L. Kocarev. Chaos-based cryptography :a brief overview. *IEEE Circuits and Systems Magazine*, 1(3) :6–21, 2001.

[Lan02] S. Lang. *Algebra, Graduate Texts in Mathematics*. Berlin,New York : Springer-Verlag, 2002.

[MAD08] G. Millérioux, J. M. Amigó, and J. Daafouz. A connection between chaotic and conventional cryptography. *IEEE Trans. on Circuits and Systems I : Regular Papers*, 55(6), July 2008.

[Mau91] U. M. Maurer. New approaches to the design of self-synchronizing stream cipher. *Advance in Cryptography, In Proc. Eurocrypt '91, Lecture Notes in Computer Science*, pages 548–471, 1991.

[MBA+03] G. Millérioux, G. Bloch, J. M. Amigo, A. Bastos, and F. Anstett. Real-time video communication secured by a chaotic key stream cipher. In *Proc. of IEEE 16th European Conference on Circuits Theory and Design, ECCTD'03*, pages 245–248, Krakow, Poland, September 1-4 2003.

[MD03] G. Millérioux and J. Daafouz. An observer-based approach for input independent global chaos synchronization of discrete-time switched systems. *IEEE Trans. Circuits. Syst. I : Fundamental Theo. Appl*, 50(10) :1270–1279, October 2003.

[MD04] G. Millérioux and J. Daafouz. Unknown input observers for message-embedded chaos synchronization of discrete-time systems. *International Journal of Bifurcation and Chaos*, 14(4) :1357–1368, April 2004.

[MOV96] A. J. Menezes, P. C. Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, October 1996.

[MVH93a] Cuomo K. M., Oppenheim A. V., and Strogatz S. H. Robustness and signal recovery in a synchronized chaotic system. *International Journal of Bifurcation and Chaos*, 3(6) :1629–1638, 1993.

[MVH93b] Cuomo K. M., Oppenheim A. V., and Strogatz S. H. Synchronization of lorenz-based chaotic circuits with applications to communications. *IEEE Trans. Circuits. Syst. II : Anal. Digit. Sign. Process*, 40(10) :626–633, 1993.

[PC90] L.M. Pecora and T.L. Carroll. Synchronization in chaotic systems. *Phys. Rev. Lett.*, 64 :821–824, 1990.

[PM01] Palaniyandi P. and Lakshmanan M. Secure digital signal transmission by multistep parameter modumation and alternative driving of transmitter variables. *International Journal of Bifurcation and Chaos*, 11(7) :2031–2036, 2001.

[TP99] R. H.C. Takahashi and P. L. D. Peres. Unknown input observers for uncertain systems : a unifying approach. *European Journal of Control*, 5(2) :261–275, 1999.

[UMW96] Feldmann U., Hasler M., and Schwarz W. Communication by chaotic signals :the inverse system approach. *Int. J. of Circuit Theory Appl.*, 24 :551–579, 1996.

[UOL+93] Parlitz U., Chua L. O., Kocarev L., Halle K. S., and Shang A. Transmission of digital signals by chaotic synchronization. *International Journal of Bifurcation and Chaos*, 3(2) :973–977, 1993.

[VMS03] R. Vidal, Y. Ma, and S. Sastry. An algebraic geometric approach to the identification of a class of linear hybrid systems. *42nd IEEE Conference on Decision and Control 2003 (CDC'03)*, 2003.

[Wan91] D. Wang. Elimination theory, methods and practice. *Mathematics and Mathematics-Mechanization*, pages 91–137, 1991. available at http ://www-calfor.lip6.fr/ wang/.

[WO93] Wu C. W. and Chua L. O. A simple way to synchronize chaotic systems with applications to secure communications systems. *International Journal of Bifurcation and Chaos*, 3(6) :1619–1627, 1993.

[Yan04] T. Yang. A survey of chaotic secure communication systems. *Int. J. of Computational Cognition*, 2004. (available at `http://www.YangSky.com/yangijcc.htm`).

INSTITUT NATIONAL
POLYTECHNIQUE
DE LORRAINE

AUTORISATION DE SOUTENANCE DE THESE
DU DOCTORAT DE L'INSTITUT NATIONAL
POLYTECHNIQUE DE LORRAINE

**o0o**

VU LES RAPPORTS ETABLIS PAR :
**Monsieur Nacim RAMDANI, Maître de Conférences, LIRMM, Université de Montpellier 2, Montpellier**

**Monsieur Krishna BUSAWON, Professeur, Northumbria University, United Kingdom**

Le Président de l'Institut National Polytechnique de Lorraine, autorise :

**Monsieur VO TAN Phuoc**

NANCY BRABOIS
2, AVENUE DE LA
FORET-DE-HAYE
BOITE POSTALE 3
F - 54501
VANDŒUVRE CEDEX

à soutenir devant un jury de l'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE, une thèse intitulée :

**"Théorie du contrôle et systèmes hybrides dans un contexte cryptographique"**

en vue de l'obtention du titre de :

DOCTEUR DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE

Spécialité : « **Automatique et Traitement du Signal** »

Fait à Vandoeuvre, le 02 novembre 2009
Le Président de l'I.N.P.L.,
F. LAURENT

TEL. 33/03.83.59.59.59
FAX. 33/03.83.59.59.55

**Théorie du contrôle et systèmes hybrides dans un contexte cryptographique**

**Résumé :** La thèse traite de l'utilisation des systèmes hybrides dans le contexte particulier des communications sécurisées et de la cryptographie. Ce travail est motivé par les faits suivants. L'essor considérable des communications qui a marqué ces dernières décennies nécessite des besoins croissants en terme de sécurité des échanges et de protection de l'information. Dans ce contexte, la cryptographie joue un rôle central puisque les informations transitent la plupart du temps au travers de canaux publics. Parmi les nombreuses techniques de chiffrement existants, le chiffrement par flot se distingue tout particulièrement lorsqu'on le débit d'une communication sécurisée est privilégié. Les chiffreurs par flot sont construits à partir de générateurs de séquences complexes décrits par des systèmes dynamiques et devant être synchronisés de part et d'autre du canal d'échanges. Les objectifs et les résultats de ce travail se déclinent en trois points. Tout d'abord, l'intérêt d'utiliser des systèmes hybrides en tant que primitives cryptographiques est motivé. Par la suite, une étude comparative est menée afin d'établir une connexion entre les algorithmes de masquage de l'information basés sur le chaos et les algorithmes de chiffrement usuels. L'étude porte exclusivement sur des considérations structurelles et repose sur des concepts de la théorie du contrôle, en particulier l'inversibilité à gauche et la platitude. On montre que la technique de masquage dite par inclusion, qui consiste à injecter l'information à protéger dans une dynamique complexe, est la plus efficace. De plus, on montre que sous la condition de platitude, un système de masquage par inclusion est structurellement équivalent à un chiffreur par flot particulier appelé auto-synchronisant. Enfin, des méthodes de cryptanalyse pour évaluer la sécurité du masquage par inclusion sont proposées pour une classe particulières de systèmes hybrides à savoir les systèmes linéaires à commutations. A nouveau, des concepts de la théorie du contrôle sont utilisés, il s'agit de l'identifiabilité paramétrique et des algorithmes d'identification. Des spécificités relatives au contexte particulier de la cryptographie sont prises en compte. En effet, contrairement à la plupart des cas rencontrés dans le domaine du contrôle où les variables des modèles dynamiques sont continues car relatives à des systèmes physiques, les variables prennent ici des valeurs discrètes. Les modèles dynamiques sont en effet décrits non plus dans le corps des réels mais dans des corps finis en vue d'une implémentation sur des machines à états finis tels ordinateur ou tout autre dispositif numérique.

**Mots clés :** systèmes hybrides, dynamiques complexes, synchronisation, chiffrement, cryptanalyse.

**Control theory and hybrid systems in a cryptographical context**

**Abstract:** This manuscript deals with a specific engineering application involving hybrid dynamical systems : secure communications and cryptography. The work is motivated by the following facts. The considerable progress in communication technology during the last decades has led to an increasing need for security in information exchanges. In this context, cryptography plays a major role as information is mostly conveyed through public networks. Among a wide variety of cryptographic techniques, stream ciphers are of special interest for high speed encryption. They are mainly based on generators of complex sequences in the form of dynamical systems, which must be synchronized at the transmitter and receiver sides. The aim of this work is threefold. First, the interest of resorting to hybrid dynamical systems for the design of cryptographic primitives is motivated. Secondly, a connection between chaotic and conventional cryptography is brought out by comparing the respective algorithms proposed in the open literature. The investigation focuses on structural consideration. Control theoretical concepts, in particular left invertibility and flatness, are the central tools to this end. It is shown that the so-called message-embedding technique, consisting in injecting the information to be concealed into a dynamical system, is the most relevant technique. Furthermore, it is shown that, under the flatness condition, the resulting cipher acts as a self-synchronizing stream cipher. Finally, cryptanalytic methodologies for assessing the security of the message-embedded cryptosystem involving a special class of hybrid systems, namely the switched linear systems, are proposed. Again concepts borrowed from control theory, namely identifiability and identification, are considered. Specificities related to the context are taken into account. The variables describing the dynamical systems do not take values in a continuum unlike what usually happens in automatic control when physical models are considered. They rather take values in finite cardinality sets, especially finite fields, since an implementation in finite state machines, say computers or digital electronic devices, is expected.

**Keywords :** hybrid systems, complex dynamics, synchronization, cryptography, cryptanalysis