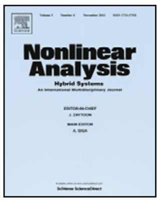
Nonlinear Analysis: Hybrid Systems 7 (2013) 68–79

Contents lists available at SciVerse ScienceDirect

Nonlinear Analysis: Hybrid Systems

journal homepage: www.elsevier.com/locate/nahs



Designing self-synchronizing switched linear systems: An application to communications

Jérémy Parriaux *, Gilles Millérioux

Université de Lorraine, Research Center for Automatic Control of Nancy (CRAN UMR 7039), 2 rue Jean Lamour, 54519 Vandoeuvre-les-Nancy, France

A R T I C L E I N F O

Keywords:
Switched systems
Synchronization
Communication
Flatness
Invertibility

A B S T R A C T

This paper addresses the problem of self-synchronizing dynamical systems in a so-called master-slave configuration. The study is motivated by potential cryptographic applications. It is shown that the notion of flatness is central for guaranteeing a finite-time self-synchronization and that the concept of transmission zero plays also an important role. Next, the finite-time synchronization is relaxed to give rise to a so-called statistical self-synchronization, a mode of operation which makes sense in classical cryptography which operates over finite fields. The fact that switched linear systems are of great interest in this context is motivated.

1. Introduction

Synchronization of dynamical systems is an important purpose in many fields like biology, mechanics, communications. Synchronization means coordinated behavior of different interconnected entities involved in an overall system. Many different definitions and related configurations, in terms of coupling, can be investigated. An exhaustive and interesting overview can be found in [1]. A special kind of synchronization is the self-synchronization. By self-synchronization, it is meant a coordinated behavior which is achieved without any external control.

The configuration under consideration in this paper is a master–slave configuration with unidirectional coupling. It is borrowed from the field of communications and more specifically secure transmissions. In this context, cryptography plays a central role. It is the discipline which is mainly intended to protect information and to guarantee confidential exchanges through public channels. Since the 90's, many "scrambling" methods resorting to synchronized chaotic dynamical systems have been proposed. In the works [2–5], it is highlighted the connection between cryptography and the use of synchronized dynamical systems in a master–slave configuration exhibiting complex dynamics. The exogenous input of the master dynamical system is the information to be encrypted. The master plays the role of the encryptor. The slave is a dynamical system which plays the role of the decryptor. The coupling is achieved through the output of the encryptor which acts as the cryptogram. In all the "scrambling" methods, synchronization between the master and the slave is guaranteed without any external control. In other words, self-synchronization is achieved by means of the output coupling. That corresponds to some classical communication setups for which insertion of synchronization flags in the transmitted packets is forbidden for throughput purposes.

As it turns out, most of the "scrambling" methods resort to observer-based approaches for ensuring the synchronization and in general, asymptotical synchronization is achieved. It is clear that the asymptotical convergence may appear when operating over the field of real numbers as it is the case for chaotic systems. On the other hand, it makes no longer sense when operating over finite fields as it is precisely the case in classical cryptography. In [3], it has been shown that a finite-time synchronization can be achieved whenever the dynamical system playing the role of the encryptor is flat. In this case,

* Corresponding author.
E-mail addresses: jeremy.parriaux@esstin.uhp-nancy.fr (J. Parriaux), gilles.millerioux@esstin.uhp-nancy.fr (G. Millérioux).

1751-570X/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.nahs.2012.05.001

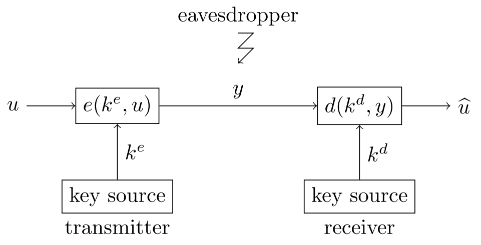


Fig. 1. General encryption mechanism.

the communication scheme is structurally equivalent to a so-called classical *self-synchronizing stream cipher*. As a result, resorting to flat dynamical systems not only makes sense from a cryptographic point of view but would provide a new approach for the design of *self-synchronizing stream cipher*s. However, this study was reduced to analysis.

The aim of the present work is to provide a constructive approach for the design of dynamical systems having the self-synchronization property. Discrete-time switched linear systems are specifically addressed because they can be related to the so-called Maiorana–McFarland construction which has proved to produce functions that have many interesting cryptographic properties (see [6]). Then, the finite-time convergence will be relaxed to give rise to the so-called statistical self-synchronizing stream ciphers. Such ciphers still make sense in classical cryptography but they have only been touched on so far. Hence, the constructions proposed in this paper can be considered as a first step towards a complete framework for designing new classes of self-synchronizing stream ciphers. Let us point out that we mainly focus on the structural considerations of the ciphers while disregarding the security aspects which would be out of the scope here and are discussed in companion papers.

The outline of this paper is the following. In Section 2, strict necessary background on cryptography is provided. The role of self-synchronization in this context is emphasized and a formal definition of finite-time self-synchronization is given. In Section 3, the design of admissible master–slave configurations, described by piecewise linear systems, achieving finite-time self-synchronization is detailed. A constructive approach for guaranteeing the self-synchronization is suggested. It is mainly based on the notion of nilpotent semigroups. A connection between the issue of guaranteeing self-synchronization and the concept of flatness is brought out. Further considerations for the design are developed in Section 4 where it is shown that the concept of transmission zero of a dynamical system plays an important role as well. Section 5 extends finite-time self-synchronization to statistical self-synchronization by releasing some constraints. Finally, Section 6 is devoted to illustrative examples.

Notation. $\mathbf{1}_n$ stands for the identity matrix of dimension n, $\mathbf{0}$ stands for the zero matrix of appropriate dimension regarding the situation. We denote by $\{z\}_{k_1}^{k_2}$ the sequence $\{z_{k_1}, \ldots, z_{k_2}\}$ when the initial and final times k_1 and k_2 are defined, otherwise the sequence is merely denoted by $\{z\}$.

2. Cryptography and synchronization

2.1. Background on cryptography

A general encryption mechanism, also called cryptosystem or cipher, is depicted in Fig. 1. We are given an alphabet A that is, a finite set of elements named symbols. On the *transmitter* part, a plaintext (also called information or message) $\{u\} \in \mathcal{U}$ ($\mathcal{U}$ is called the message space) consisting of a string of symbols $u_k \in A$ 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 $\{y\} \in \mathcal{C}$ ($\mathcal{C}$ is called the ciphertext space), a string of symbols $y_k \in B$, B being a set usually identical to A, is conveyed through a public channel to the *receiver*. At the receiver side, the ciphertext y_k 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. Cryptography distinguishes asymmetric and symmetric ciphers. Asymmetric cryptography is largely based upon computationally very demanding mathematical problems, for instance, integer factorization into primes. It is not discussed in this paper.

In symmetric encryption, both keys are identical that is, $k^d = k^e$. That explains the terminology "symmetric". This kind of encryption obeys a master–slave configuration. The transmitter, that is the master, is called in this context the cipher. It delivers a complex sequence (theoretically indistinguishable from a uniformly random one) used to conceal information. The information to be kept secret is, in some sense, "mixed" with the complex sequence so that the resulting sequence (called the cryptogram) conveyed to the receiver, cannot be understood by any unauthorized party. For proper information recovery, the receiver, that is the slave, called in this context the decryptor, must deliver the same complex sequence synchronized with the encryptor.

For stream ciphers depicted in Fig. 2, the keys k^e and k^d are replaced by time-varying sequences called *running keys* or *key-streams*. They are denoted by $\{x\}$ (with samples x_k) at the transmitter part and by $\{\widehat{x}\}$ (with samples $\widehat{x}_k$) at the receiver part. As a result, stream ciphers require key-stream generators at both ends. The key-streams $\{x\}$ and $\{\widehat{x}\}$ must be synchronized