Available online at www.sciencedirect.com
ScienceDirect
Future Generation Computer Systems 23 (2007) 727–733
FGCS
FUTURE GENERATION COMPUTER SYSTEMS
www.elsevier.com/locate/fgcs

Symbolic/numeric analysis of chaotic synchronization with a CAS

A. Gálvez*, A. Iglesias

Department of Applied Mathematics and Computational Sciences, University of Cantabria, Avda. de los Castros s/n, E-39005, Santander, Spain

Received 29 September 2006; accepted 17 October 2006
Available online 29 December 2006

Abstract

The synchronization of chaotic dynamical systems has received increased attention during the last few years, mostly because of its potential applications to secure communications. However, the computational analysis of this issue is still a challenge. In this paper we perform a symbolic/numeric analysis of the chaotic synchronization by using the Computer Algebra System (CAS) *Mathematica*.

Keywords: Synchronization; Dynamical systems; Computer algebra systems (CAS); Mathematica

1. Introduction

Nowadays, the use of coupled and synchronized systems is quite standard in several fields, ranging from physical systems [3,4] to computer science [2,14], etc. However, the possibility of synchronizing chaotic systems is not so intuitive, since these systems are very sensitive to small perturbations on the initial conditions and, therefore, close orbits of the system quickly become uncorrelated [15]. Surpringly, in 1990 it was shown that certain subsystems of chaotic systems can be synchronized by linking them with common signals [16]. In particular, the authors reported the synchronization of two *identical* (i.e., two copies of the same system with the same parameter values) chaotic systems. They also showed that, as the differences between those system parameters increase, synchronization is lost. Since then, the synchronization of chaotic systems has been extensively investigated both from the theoretical [17] and the experimental [3,4] points of view. In addition, some possible applications to various fields, such as secure communications, have been discussed [4,5,10].

Although the theory of dynamical systems is not new, only with the advent of the computers have we been able to simulate those systems and to capture the subtle details of their otherwise unpredictable behavior. Because of the strong sensitivity to initial conditions property, the numerical routines are very prone to errors and cannot describe accurately and completely the nonlinear nature of dynamical systems [9]. This makes the computer algebra systems (CAS) – such as *Mathematica* or *Maple* – indispensable tools in this field.

Some years ago, one of the authors published some work on dynamical systems by using the CAS *Mathematica* [8]. The experience was very positive and convinced us that *Mathematica* is a very convenient tool for these purposes. In this paper, we perform a symbolic/numeric analysis of the chaotic synchronization phenomena by using *Mathematica*.

2. Chaotic synchronization

We start our discussion by loading the package:

```
In[1]:=<<DynamicalSystems`ChaosSynchronization`
```

2.1. Pecora–Carroll scheme for chaotic synchronization

In 1990 Pecora and Carroll [16] showed that when a state variable from a chaotic system is input into a replica subsystem of the original one, both systems can be synchronized identically. In mathematical terms, given a couple of autonomous n-dimensional identical chaotic systems $\dot{x_1} = f(x_1)$ and $\dot{x_2} = f(x_2)$ as the drive and response systems respectively, the basic idea of the Pecora–Carroll (PC) scheme is decomposing the drive system into two subsystems,

* Corresponding author.
E-mail addresses: akemi.galvez@postgrado.unican.es (A. Gálvez), iglesias@unican.es (A. Iglesias).
URL: http://personales.unican.es/iglesias (A. Iglesias).

0167-739X/$ - see front matter
doi:10.1016/j.future.2006.10.010

$\dot{x_1} = (\dot{u_1}, \dot{v_1})$, with $x_1 \in \mathbb{R}^n$, $u_1 \in \mathbb{R}^p$ and $v_1 \in \mathbb{R}^q$ (where $n = p + q$) as:

$$\left.\begin{aligned}\dot{u_1} &= g(u_1, v_1)\\ \dot{v_1} &= h(u_1, v_1)\end{aligned}\right\} \text{ drive,} \tag{1}$$

and considering one of the decomposed subsystems as the driving signal, say u_1, to be injected into the response system:

$$\dot{v_2} = h(u_1, v_2)\} \text{ response,} \tag{2}$$

where u_1 is the set of connecting variables. Note that the system (1) is independent of the response system, whereas (2) is driven by u_1 (*unidirectional coupling*). The scheme given by Eqs. (1) and (2) can be generalized by considering a response system given by $\dot{v_2} = k(u_1, v_2)$, that is, by assuming that the functions h and k describing the dynamics of $\dot{v_1}$ and $\dot{v_2}$ respectively must not be the same. This situation is usually referred to as *heterogeneous* (or *inhomogeneous*) *driving*.

To illustrate the PC synchronization method, we consider the well-known Lorenz and Rössler systems, described respectively by:

$$\begin{cases} x' = \sigma(y - x)\\ y' = (r - z)x - y\\ z' = xy - bz \end{cases} \quad \text{and} \quad \begin{cases} x' = -y - z\\ y' = x + ay\\ z' = b + z(x - c). \end{cases}$$

Both can be efficiently represented in *Mathematica* as:

```
In[2]:=Lorenz[x_,y_,z_]:=
        {σ (y-x),(r-z)x-y,x y-b z};
In[3]:=Rossler[x_,y_,z_]:= {-y-z,x+a y,b+z (x-c)};
```

Given a dynamical system and the list of its variables, the `Parameters` command returns all the system parameters:

```
In[4]:=Parameters[#[x,y,z],{x,y,z}]& /@
        {Lorenz,Rossler}
```

Out[4] := $\{\{b, r, \sigma\}, \{a, b, c\}\}$.

It is useful to write a dynamical system, X, as a linear system by taking the first-order approximation, $\dot{X} = J(X).X$, where the square matrix $J(X)$ is called the *Jacobian matrix* of the system. The `JacobianMatrix` command calculates the Jacobian matrix of a dynamical system with respect to its list of variables. For example:

```
In[5]:=JacobianMatrix[Lorenz[x,y,z],{x,y,z}]
```

$$Out[5] := \begin{pmatrix} -\sigma & \sigma & 0\\ r - z & -1 & -x\\ y & x & -b \end{pmatrix}.$$

With this system viewed as the transmitter or master system, we introduce the drive signal y which can be used at the receiver or slave system, to perform chaotic synchronization. In other words, we apply the PC synchronization scheme (1) and (2) of two Lorenz systems given by $u_1 = y_1$ and $v_1 = (x_1, z_1)$. In order to determine if the synchronization is achieved, we consider the Jacobian matrix of the error dynamics between the slave and the master systems, the so-called *Jacobian Conditional Matrix* (JCM). For a given input including: (1) a dynamical system, (2) its list of variables, and (3) a specific connection given by the driving variable/s, the `JCMatrix` command returns the JCM of that connection. For instance:

```
In[6]:=JCMatrix[Lorenz[x,y,z],{x,y,z},y]
```

$$Out[6] := \begin{pmatrix} -\sigma & 0\\ y & -b \end{pmatrix}.$$

It can be proved that the master and slave systems will synchronize if the eigenvalues of the JCM are all negative [17]. In general, those eigenvalues depend not only on the given dynamical system and the injected variable, but also on the parameter values of the system:

```
In[7]:=Eigenvalues[%]
```

Out[7] := $\{-\sigma, -b\}$.

In their original paper, Pecora and Carroll considered the parameter values:

```
In[8]:= param={b- > 8/3,r- > 60, σ - > 10};
```

which yield a chaotic behavior, as shown in Fig. 1 (left):

```
In[9]:= NDSolve[{x'[t]==σ*(y[t]-x[t]),x[0]==4,
    y'[t]==(r-z[t])*x[t]-y[t],y[0]==1,
    z'[t]==x[t]*y[t]-b*z[t],z[0]==30} /. param,
    {x,y,z},{t,0,35},Method->ExplicitRungeKutta];

In[10]:= ParametricPlot3D[Evaluate[{x[t],y[t],
   z[t]} /. %],
    {t,0,35},PlotPoints->10000,PlotRange->All,
    ViewPoint->{1.618,-2.849,0.845}];
```

Out[10] := *See Fig. 1.* (*left*).

Note that, for this choice of the system parameters, the connection in variable y is synchronizing. On the contrary, injecting variable x (i.e., taking $u_1 = x_1$ and $v_1 = (y_1, z_1)$ in (1) and (2)) leads to the JCM:

```
In[11]:=JCMatrix[Lorenz[x,y,z],{x,y,z},x]
```

$$Out[11] := \begin{pmatrix} -1 & -x\\ x & -b \end{pmatrix}$$

whose eigenvalues are:

```
In[12]:=Eigenvalues[%]
```

$$Out[12] := \left\{\frac{1}{2}\left(-b - \sqrt{b^2 - 2b - 4x^2 + 1} - 1\right), \frac{1}{2}\left(-b + \sqrt{b^2 - 2b - 4x^2 + 1} - 1\right)\right\}.$$

Because the eigenvalues do depend on the system variables, synchronization cannot be determined automatically. The solution to this question is given by the Lyapunov exponents of the difference system, since they indicate if small displacements of trajectories are along stable or unstable directions. Lyapunov exponents of the v_2-subsystem for a particular drive trajectory are called *conditional Lyapunov exponents* (CLE). If we are looking for a stable subsystem, then all the exponents must be negative so that the small perturbations will exponentially