Available online at www.sciencedirect.com
ScienceDirect
SoftwareX 3–4 (2015) 1–5
SoftwareX
www.elsevier.com/locate/softx

A practical test for noisy chaotic dynamics

Ahmed BenSaïda

LaREMFiQ - IHEC, Sousse University, B.P. 40 Route de la ceinture, Sahloul 3, Sousse 4054, Tunisia

Received 12 April 2015; received in revised form 15 June 2015; accepted 21 August 2015

Abstract

This code computes the largest Lyapunov exponent and tests for the presence of a chaotic dynamics, as opposed to stochastic dynamics, in a noisy scalar series. The program runs under MATLAB® programming language.

Keywords: Lyapunov exponent; Noisy chaos; MATLAB

Code metadata

Current code version	v5.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-15-00007
Legal Code License	BSD-3-Clause, see http://opensource.org/licenses/BSD-3-Clause
Code versioning system used	none
Software code language	MATLAB® [1]
Requirements	statistics and optimization toolboxes
Support email for questions	ahmedbensaida@yahoo.com

1. Motivation and significance

In a scientific context, the word chaos has a different meaning than it does in its general usage as a state of confusion or a total disaster. Chaos, with reference to chaos theory, refers to an apparent lack of order in a system that nevertheless obeys particular laws or rules, a condition discovered by the physicist Henri Poincaré in the early 20th century, which refers to an inherent lack of predictability in some physical systems known as dynamical instability. For example, chaos is observed in the movement of a driven pendulum where it may behave erratically and show irregular sequences of left and right turns, streams in the ocean [2,3], and in meteorological science [4,5]. Hence, a chaotic system – no matter how complex it may be – relies upon a precise order, and very small changes can cause very complex behaviors, known as *sensitive dependence on initial conditions*. This random-like and unpredictable behavior is known as *the butterfly effect*.

The butterfly effect, first described by Edward Lorenz at the December 1972 meeting of the American association for the advancement of science in Washington D.C., vividly illustrates the essential idea of chaos theory. In nonlinear dynamics theory, a butterfly flapping its wings in Brazil can produce a tornado in Texas due to nonlinearities in weather processes. The example of such a small system, as a butterfly, being responsible for creating such a large and distant outcome, as a tornado in Texas, illustrates the impossibility of making predictions for complex systems despite the fact that these are determined by underlying conditions that can never be sufficiently articulated to allow long-range predictions.

Chaotic dynamics are closely related in appearance to stochastic dynamics, and the BDS test [6] cannot separate them [7]. Hence, we need a practical test to detect chaos even when the data are noisy.

E-mail address: ahmedbensaida@yahoo.com.

http://dx.doi.org/10.1016/j.softx.2015.08.002

Theoretical basis of the code are available in [7,8], practical use of the test is performed by [9,10], among others. The reminder of the article is as follows: in Section 2 we briefly describe the code; Section 3 presents some examples; in Section 4 we discuss the impact of the code; and finally, we conclude in Section 5. Code metadata is given before Section 1.

2. Software description

Tests for chaos are scarce in the literature, and practical implementation is far from evident [11]. Moreover, to test for chaotic dynamics, we need to have pure data free from frictions engendered by measurement errors, see [12] for detailed discussion. The present code estimates the *largest Lyapunov exponent* (Lyapunov exponent henceforth) in a noisy time series [7], and decides whether the data are chaotic or stochastic based on a confidence level α. The main advantage of this test is that it can be conducted directly on experimental data without the need to define the generating equations.

The Lyapunov exponent λ measures the average exponential divergence (positive exponent) or convergence (negative exponent) rate between nearby trajectories within a time horizon that differ in initial conditions only by an infinitesimally small amount. We distinguish 3 cases of λ:

- $\lambda < 0$: the orbit attracts to a stable fixed point. A negative Lyapunov exponent is characteristic of dissipative or non-conservative system. Such a system exhibits asymptotic stability; the more negative the exponent, the greater the stability. Super-stable fixed points and super-stable periodic points have $\lambda \to -\infty$.
- $\lambda = 0$: the orbit is a neutral fixed point. A Lyapunov exponent of zero indicates that the system is in a steady-state mode or near the transition to chaos.
- $\lambda > 0$: the orbit is unstable and chaotic. Nearby points of an orbit, no matter how close, will diverge to any arbitrary separation. The larger the exponent, the more unstable the system.

To compute the Lyapunov exponent and to conduct the test given the hidden noise in the data, we need to define two components:

1. The activation function `ActiveFN` of the neural network that approximates the chaotic map [7].
2. The order (L, m, q) that defines the complexity of the neural network [7]. Choosing low parameters may prevent the neural network from reasonably approximating the map that generates the data. On the other hand, large parameters increase computational time exponentially because the number of coefficients to estimate increases. [8] suggests any triplet between (5, 6, 5) and (10, 12, 10).

2.1. Software architecture

The code is a single *m*-file that runs under MATLAB® programming language [1], all subroutines are included in the principal file chaostest.[1]

[1] We are planning to write a version that runs under R, http://CRAN.R-project.org.

2.2. Software functionalities

The code chaostest can detect the presence of chaotic dynamics. It tests the positivity of the dominant (or largest) Lyapunov exponent λ at a specified confidence level.

The test hypothesis `H` are: null hypothesis $H_0 : \lambda \geqslant 0$, which indicates the presence of chaos; and alternative hypothesis $H_1 : \lambda < 0$, which indicates the absence of chaos.

The syntax under MATLAB® command prompt is:

```
Command Window
>> [H, pVal, LAMBDA, Orders, CI] = chaostest(Series, ActiveFN, maxOrders, ALPHA)
```

Required input

`Series`—a vector of observations to test. The minimum allowed size is $(m\,L + 1)$ observations.

Optional inputs

`ActiveFN`—string containing the activation function to use in the neural network estimation. It can be:

`'tanh'`, $f(u) = \tanh u$, domain $= (-1, 1)$, this is the default;
`'logistic'`, $f(u) = \frac{1}{1+e^{-u}}$, domain = (0, 1). The logistic is not recommended due to the limited domain;
`'sigmoid'`, $f(u) = u\,\frac{1+\left|\frac{u}{2}\right|}{2+|u|+\frac{u^2}{2}}$, domain $= (-1, 1)$.

`maxOrders`—the orders (L, m, q). This must be a vector containing 3 elements, default = `[5,6,5]`.

`ALPHA`—the significance level α of the test (default = `0.05`).

Outputs

`H = 0`—accept the null hypothesis of chaos at significance level α.

`H = 1`—reject the null hypothesis of chaos at significance level α.

`pVal`—the *p*-value. Small values of `pVal` cast doubt on the validity of the null hypothesis of chaos.

`LAMBDA`—the Lyapunov exponent λ.

`Orders`—gives the triplet (L, m, q) that maximizes the Lyapunov exponent computed from all $L \times m \times q$ estimations, see [7] for further details.

`CI`—confidence interval for λ at significance level α.

The algorithm uses the Jacobian method [7], it needs the optimization and the statistics toolboxes of MATLAB® [1].

3. Illustrative examples

In this section we perform three examples. The first one for a noisy chaotic map, the second one for a stochastic random variable, and the last one for a real-life data.

3.1. The logistic map

The logistic map $x_t = \gamma\, x_{t-1}\,(1 - x_{t-1})$ is known to exhibit chaotic behavior when $3.57 \leqslant \gamma \leqslant 4$ [11]. So, in our example we set $\gamma = 4$ and a starting value $x_0 = 0.2$. Next, to the constructed pure chaotic map x_t, we add some noise ε_t, with $\varepsilon_t \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ is an independently and normally distributed random variable.

We choose different values of σ_ε and run the test in order to study the evolution of the Lyapunov exponent. In the first case we set $\sigma_\varepsilon = 0.01$. Hence, under MATLAB® command prompt, we type:

Command Window
```
>> % construct the logistic map x.
>> x = zeros(1000,1); % initialize x with 1000 observations.
>> x(1) = 0.2; % starting value.
>> for i = 2:1000
  x(i) = 4*x(i−1)*(1−x(i−1)); % fill in the remaining values.
end % now we have a pure logistic map.
>> rng('default'); % reset the random generator to defaults.
>> epsilon = normrnd(0,0.01,1000,1); % construct a random noise epsilon.
>> y = x + epsilon; % this is a noisy logistic map.
>> % now test for the presence of chaos in y.
>> [H,p,lambda,Orders] = chaostest(y,'tanh',[7,5,10]) % use the default value for ALPHA.
```

The result is `H = 0`, `p = 1`, `lambda = 1.7158`, and `Orders = [3,1,7]`.[2] Hence, the null hypothesis H_0 indicating the presence of chaos is accepted at 5% confidence level.

In the next case, we increase the variability of the noise ε_t by setting $\sigma_\varepsilon = 0.12$.

Command Window
```
>> % we still have the variable x in the workspace.
>> rng('default'); % reset the random generator to defaults.
>> epsilon = normrnd(0,0.12,1000,1); % re−construct the random noise epsilon.
>> z = x + epsilon; % this is a noisy logistic map.
>> [H,p,lambda,Orders] = chaostest(z,[],[7,5,10]) % use the default value for ActiveFN.
```

The result is `H = 0`, `p = 0.9180`, `lambda = 0.1166`, and `Orders = [1,1,6]`. The data are still chaotic at 5% confidence level. The estimated variance of the Lyapunov exponent [7, equation 5] increases due to the added noise into the data; consequently, the p-value of the exponent decreases from nearly `1` in the first case with lower noise to `0.9180` in the second case with larger noise.

Additionally, setting $\sigma_\varepsilon = 0.17$ and using exactly the same procedure yields `H = 0`, `p = 0.1677` and `lambda = -0.0437`. Although the Lyapunov exponent λ is negative, the test still accepts chaos at 5% confidence level. Increasing the amplitude of the noise further, will result in the rejection of the null hypothesis. Indeed, as mentioned by [7, p. 91], when the variance of the added noise increases, the data gradually tend to a stochastic system and the Lyapunov exponent decreases until it becomes negative. In this case, the noise envelops the chaotic map and we can no longer detect chaos.[3]

Alternatively, we can test the presence of chaos in the pure data x_t without adding the noise. In this case, `H = 0`, `p = 1`, `lambda = 1.8714`, and `Orders = [1,1,9]` for the same input `maxOrders = [7,8,10]`. The Lyapunov exponent of the pure skeleton chaotic map – noted λ_0 – is greater than the exponents of the noisy chaotic maps—noted λ_σ. Indeed, [7] shows that λ_σ tends to λ_0 as the amplitude of the noise decreases $\lim_{\sigma_\varepsilon \to 0} \lambda_\sigma = \lambda_0$.

3.2. Random variable

In the second example, we test the dynamics of an independently and normally distributed random variable ε_t with standard deviation $\sigma_\varepsilon = 1$. ε_t is expected to be stochastic.

Command Window
```
>> rng('default'); % reset the random generator to defaults.
>> epsilon = normrnd(0,1,1000,1); % construct the random variable epsilon.
>> [H,p,lambda] = chaostest(epsilon) % use default values for unspecified inputs.
```

The result is `H = 1`, `p = 5.23e-11`, and `lambda = -0.4850`. Hence, the alternative hypothesis H_1 indicating the absence of chaos is accepted at 5% confidence level. The tested data are therefore stochastic.

The Lyapunov exponent of a super-stable system, such as a random variable, is expected, in theory, to tend toward minus infinity $\lambda \to -\infty$. Nevertheless, the computed λ from chaostest is a finite number. In fact, as argued by [12], numbers generated by computers are called *pseudorandom* numbers based on a specific algorithm (RNG), because computers are deterministic machines and should not exhibit random behavior. The RNG has a starting value defined by an initial state (or seed), each time a pseudorandom number is generated, the state of the RNG changes accordingly in a pre-specified way based on the previous number. The art of computer language makes the generated pseudorandom numbers almost purely random. Therefore, the finite negative Lyapunov exponent of ε_t is a direct consequence of the art of generating pseudorandom numbers.[4]

3.3. Real data

The final example is an application to real data. We select the daily Euro–dollar (EUR/USD) exchange rate returns – logarithmic difference – collected from OANDA database,[5] and the daily rainfall in millimeter on Sydney, Australia collected from the Australian government bureau of meteorology.[6] Both samples start from January 1, 2005 until April 30, 2015 and plotted in Fig. 1. The file `mmc1.mat` in supplementary data contains two variables `eurusd` and `rainfall` (see Appendix A). Under MATLAB®, we type:

Command Window
```
>> load data % load the data file, which contains two variables 'eurusd' and 'rainfall'.
>> [H,p,lambda,Orders] = chaostest(eurusd) % test the dynamics of EUR/USD returns.
```

The result is `H = 1`, `p = 0`, `lambda = -0.2671`, and `Orders = [1,6,1]`. Hence, the EUR/USD exchange rate returns are stochastic. Next, to test the dynamics of the second variable `rainfall` with the sigmoid activation function, we type:

Command Window
```
>> % the variable 'rainfall' should be in the worksapce, if not load the data.
>> [H,p,lambda,Orders] = chaostest(rainfall,'sigmoid') % select the sigmoid ActiveFN.
```

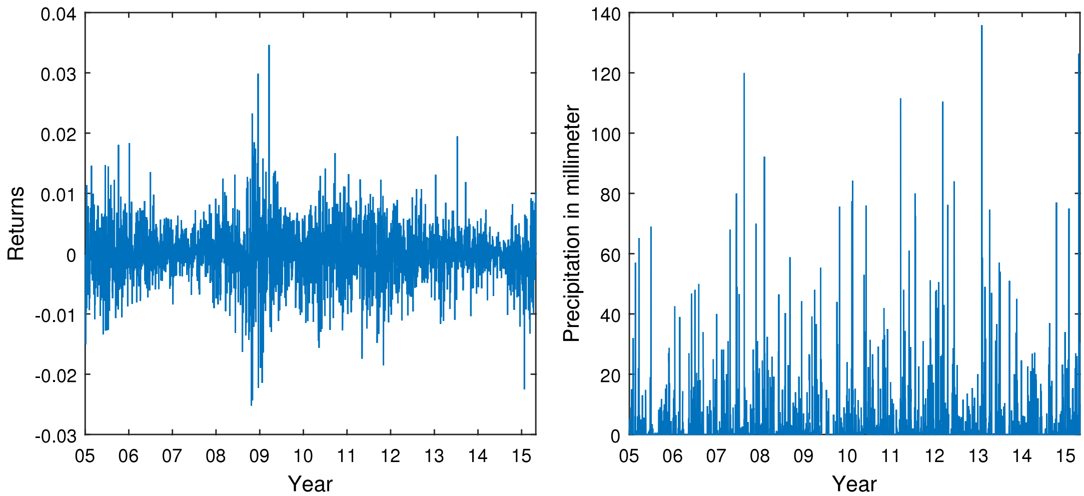
[2] Results may differ if we change the random generator state, due to the presence of the noise ε_t. For presentation purpose, we report the p-value and λ with four significant digits after the decimal point.

[3] We suggest further theoretical development to compute the limiting σ_ε beyond which the test rejects chaos.

[4] The default RNG of MATLAB ® is based on the Mersenne-Twister algorithm [13] with an initial seed that equals zero.

[5] http://www.oanda.com.

[6] http://www.bom.gov.au/climate/data.



(a) Daily EUR/USD exchange rate returns. (b) Daily rainfall on Sydney, Australia.

Fig. 1. Real data example.

We obtain `H = 0`, `p = 1`, `lambda = 1.4605`, and `Orders = [3,5,1]`. Hence, the dynamics of Sydney's rainfall is chaotic.

4. Impact

Recognizing and quantifying chaos in time series is the subject of many studies. In fact, several approaches are proposed, including estimating fractal dimensions, nonlinear forecasting, estimating entropy, and estimating Lyapunov exponents [12].

Among the methods proposed, fractal dimension estimation is the simplest one. It provides a test about the finite dimensionality of a system. However, the dimension estimates is highly sensitive to measurement error in the data and may get worse with dynamical noise. Similar difficulty exists in the entropy estimates. Nonlinear forecasting is a more general concept because it includes nonlinearity in both deterministic and stochastic systems, and it can be detected by the BDS test [6]; yet, it cannot distinguish between chaotic behavior and stochastic behavior. The problem encountered in fractal dimension and entropy estimations is avoided in the Lyapunov exponent approach.

Chaos tests currently available in the literature need noise-free data, since any measurement error causes the systematic rejection of chaos [11]. Moreover, practical implementation of these tests is challenging. The chaostest code is powerful in detecting chaotic dynamics, as opposed to stochastic dynamics, even in presence of moderate noises. This feature makes it appealing to physicists, meteorologist, financial analysts, and other scientists who study nonlinear dynamics and complex phenomena. The effectiveness of the test has been shown in a previous study [7].

The chaostest code is applied on real life data. For example, [7] tested the presence of chaotic dynamics in six daily exchange rates and six daily world indexes, and found that these financial data are stochastic. [8] tested the dynamics of the intra-daily American index S&P 500 over different frequencies, and found no evidence of chaos. [14] tested the dynamics of the inflation in the United States using long-range monthly and annual data, they rejected the existence of chaos in both data. [10] applied the test on high-frequency returns in order to detect non-linearity in the Athens composite share price index. [9] used the test to study the dynamics of intra-day S&P 500 using a wavelet transform to improve the forecasting accuracy. Applications on physical, meteorological and oceanographic data are still in progress, and the example on Sydney's rainfall is an incentive to study other data type.

Finally, the present code can be applied on any scalar experimental data. It may be of interest to those who study nonlinear dynamical systems to determine the nature of the data: stochastic or chaotic.

5. Conclusion

The developed code can separate between stochastic and chaotic dynamics even in presence of a moderate noise. It helps researchers in physical, meteorological, financial and any other field to test their data for the presence of chaos.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.softx.2015.08.002.

References

[1] The MathWorks, Inc., MATLAB®—The Language of Technical Computing, Natick, Massachusetts (2015). URL http://www.mathworks.com/products/matlab.
[2] Dijkstra HA. Nonlinear physical oceanography. 2nd ed. Netherlands: Springer; 2005.
[3] Dijkstra HA, Ghil M. Low-frequency variability of the large-scale ocean circulation: A dynamical systems approach. Rev Geophys 2005;43:1–38.
[4] Ghil M, Kimoto M, Neelin JD. Nonlinear dynamics and predictability in the atmospheric sciences. Rev Geophys 1991;29(Supplement):46–55.
[5] Lorenz EN. Designing chaotic models. J Atmos Sci 2005;62(5):1574–87.
[6] Brock W, Dechert D, Scheinkman J, LeBaron B. A test for independence based on the correlation dimension. Econometric Rev 1996;15(3):197–235.

[7] BenSaïda A, Litimi H. High level chaos in the exchange and index markets. Chaos Solitons Fractals 2013;54:90–5.
[8] BenSaïda A. Noisy chaos in intraday financial data: Evidence from the American index. Appl Math Comput 2014;226:258–65.
[9] Lahmiri S. Improving forecasting accuracy of the S&P 500 intra-day price direction using both wavelet low and high frequency coefficients. Fluct Noise Lett 2014;13(1):1450008.
[10] Anagnostidis P, Emmanouilides CJ. Nonlinearity in high-frequency stock returns: Evidence from the Athens stock exchange. Phys A 2015;421:473–87.
[11] Gottwald GA, Melbourne I. Testing for chaos in deterministic systems with noise. Phys D 2005;212:100–10.
[12] BenSaïda A. Are financial markets stochastic: A test for noisy chaos. Amer Internat J Contemp Res 2012;2(8):57–68.
[13] Matsumoto M, Nishimura T. Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator. ACM Trans Model Comput Simul 1998;8(1):3–30.
[14] Plakandaras V, Gogas P, Gupta R, Papadimitriou T. US inflation dynamics on long-range data. Appl Econom 2015;47(36):3874–3890.