ELSEVIER Applied Mathematics and Computation 114 (2000) 205–214

APPLIED MATHEMATICS AND COMPUTATION

www.elsevier.com/locate/amc

A note on the computation of the largest p LCEs of discrete dynamical systems

Firdaus E. Udwadia [a,*], Hubertus F. von Bremen [b], Wlodek Proskurowski [c]

[a] *Department of Mechanical Engineering, Civil Engineering and Decision Systems, University of Southern California, CA 90089-1453, USA*

[b] *Department of Mechanical Engineering, University of Southern California, CA 90089-1453, USA*

[c] *Department of Mathematics, University of Southern California, CA 90089-1113, USA*

Abstract

Two efficient and numerically stable methods for the computation of the largest p Lyapunov characteristic exponents of an n dimensional discrete dynamical systems are presented. The efficiencies of the proposed methods are compared with the efficiencies of other methods through an operation count, and illustrated with an example. © 2000 Elsevier Science Inc. All rights reserved.

Keywords: Computation of Lyapunov Characteristic Exponents; Discrete dynamical systems; Largest p LCEs; Numerically stable; Efficient

1. Introduction

A positive Lyapunov Characteristic Exponent (LCE) of a nonlinear dynamical system is often used as an indicator of chaos. If one is interested in determining whether a dynamical system is chaotic or not, often just a few of the largest LCEs may provide the answer. If just the largest p LCEs provide the needed information about the dynamical system, it may be redundant to try to compute all of them. The issue of computing just the largest p LCEs is addressed by Benettin et al. [1], Geist et al. [2], and Dieci and Van Vleck [3]. It is assumed in this note that the dynamical systems under consideration are such

[*] Corresponding author.

PII: S0096-3003(99)00113-7

that the LCEs form a monotonically decreasing sequence, which implies that it is possible to compute just the largest p LCEs.

Since the computation of the largest p LCEs of a discrete dynamical system is computationally less expensive then the computation of all the LCEs, this note deals with the adaptation of the HRQB method provided by von Bremen et al. [4] (which computes all the LCEs of a discrete dynamical system) to the computation of only the largest $p \leqslant n$ LCEs. Two methods, herein called HQRBp1 and HQRBp2, are described.

2. Computation of the largest p LCE's

Consider the autonomous discrete dynamical system

$$x_{i+1} = f_i(x_i), \quad i = 0, 1, 2, \ldots, x_0 \quad \text{given} \quad \text{and} \quad x_i \in \mathscr{R}^n. \tag{1}$$

The associated linear variational equations are

$$Y_{i+1} = Df(x_i)Y_i = J_i Y_i, \quad Y_i \in \mathscr{R}^{n\times n}, \qquad i = 0, 1, 2, \ldots, Y_0 = I \tag{2}$$

provided the $\{f_i\}$ are differentiable and $J_i = Df(x_i) = (\partial f_i/\partial x)_{x_i}$. A fundamental solution of Eq. (2) is $Y^i = J_{i-1} \cdots J_0$, with $(Y^0)^{\mathrm{T}} Y^0 = I$. Thus the positive definite matrix $\Lambda = \lim_{i\to\infty} (Y^{i^{\mathrm{T}}} Y^i)^{1/(2i)}$ exists and the logarithms of its eigenvalues are the Lyapunov Characteristic Exponents denoted by $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n$. If the LCEs are monotonically decreasing, then it is possible to find the largest p LCEs by considering a reduced system of Eq. (2) (see [1–3]). The reduced rectangular system is given by

$$Y_{i+1} = J_i Y_i, \quad Y_i \in \mathscr{R}^{n\times p}, \qquad i = 0, 1, 2, \ldots, Y_0^{\mathrm{T}} Y_0 = I. \tag{3}$$

When computing the largest p LCEs, at each iteration we perform a QR-factorization of a matrix and the product of a tangent map times the Q matrix obtained from the factorization. The essential difference between computing all the LCEs and the largest p of them, lies in the size of the matrices on which we operate. In the case of all the LCEs, all matrices are square and n by n. For the case of the largest p LCEs the QR factorization is performed on an n by p rectangular matrix, thus the resulting Q is n by p and R is p by p. Based on these observations, we present the following two adaptations of the HQRB method [4].

2.1. The HQRBp1 method

The HQRBp1 method is essentially the same as the HQRB method [4], except that only p reflectors are computed; also, in the process of finding the action of the tangent map on the reflector matrices at the very last step only the first p columns of the product is computed. Using the same notation as in [4], the action of J on the first p reflector matrices is given by

$$JQ = JH^{(1)}H^{(2)} \cdots H^{(p)}, \tag{4}$$

where $H^{(i)}$ is the ith reflector matrix. Since only the first p columns of the product JQ are needed, in HQRBp1 we first find the product $J^{(p-1)} = JH^{(1)}H^{(2)} \cdots H^{(p-1)}$ using the action of the matrices on the reflector matrices without explicitly computing the reflector matrices (as done in [4] for the HQRB). Finally, the last product $J^{(p-1)}H^{(p)}$ is performed so that only the first p columns are determined. As in the HQRB method [4], the matrix $H^{(p)}$ is not explicitly computed. This approach is efficient when p is close to n. However, for the case when the ratio p/n is small, the following new and more efficient implementation is derived.

2.2. The HQRBp2 method

The difference between the HQRBp1 and the HQRBp2 methods is in the computation of the action of the next tangent map on the p reflector matrices. Taking the transpose of Eq. (4) we get, using the symmetry of the reflector matrices $H^{(i)}$

$$Q^{\mathrm{T}}J^{\mathrm{T}} = {H^{(p)}}^{\mathrm{T}} \cdots {H^{(2)}}^{\mathrm{T}}{H^{(1)}}^{\mathrm{T}}J^{\mathrm{T}} = H^{(p)} \cdots H^{(2)}H^{(1)}J^{\mathrm{T}}. \tag{5}$$

Since only the first p columns of the product JQ is called for, only the first p rows of the product $Q^{\mathrm{T}}J^{\mathrm{T}}$ is needed. From Eq. (5), the first p rows of $Q^{\mathrm{T}}J^{\mathrm{T}}$ can be computed by multiplying the first p rows of $H^{(p)}$ times all of $H^{(p-1)}$, then taking the resulting p by n matrix and multiplying it with $H^{(p-2)}$, and so on until the product is complete. Each $H^{(i)}$ is a reflector matrix, and does not need to be computed explicitly. Since the reflector $H^{(i)}$ is given by $H^{(i)} = I - w^{(i)}{w^{(i)}}^{\mathrm{T}}$, where $w^{(i)}$is a column vector, then $BH^{(i)} = B - Bw^{(i)}{w^{(i)}}^{\mathrm{T}}$. The product $Bw^{(i)}{w^{(i)}}^{\mathrm{T}}$ can be obtained by first finding the product $Bw^{(i)}$ and then multiplying by ${w^{(i)}}^{\mathrm{T}}$. This saves a considerable number of computations as opposed to computing the outer product $w^{(i)}{w^{(i)}}^{\mathrm{T}}$ first, and then multiplying by B. Additional computational savings are obtained because the first $i - 1$ elements of $w^{(i)}$ are zero. The above described steps constitute the HQRBp2 method.

The order in which reflector matrices are multiplied by the next tangent matrix is different for HQRBp1 and HQRBp2. As a consequence, very small differences (caused by round-off) in the resulting p columns of JQ could occur producing differences (of the order of the machine precision) in the upper triangular matrix R, after the QR-factorization of JQ. If the system is close to singular, at least one of the diagonal elements of R is then close to zero. The logarithms of the diagonal elements of R are used to compute the LCEs; the logarithm function being singular at zero, small changes in the diagonal elements of R will cause large changes in the logarithms of these elements. Thus for systems that are near singular, the differences in the very small LCEs between HQRBp1 and HQRBp2 could be larger than machine precision. For example, for the ill-conditioned Hilbert matrix of order $n = 10$ (with a condition number of about $10^{16} \approx 1/eps$, eps being the machine precision for Matlab), the largest difference in LCEs between HQRBp1 and HQRBp2, at the end of the third iteration, is of the order of 10^{-5}, and occurs for the smallest LCE. On the other hand, for the largest LCE the difference is of the order of *eps*.

The classical Gram–Schmidt (CGS) QR-factorization is not as stable as the HQRB, as shown in [4], and elsewhere. Nevertheless, for the sake of completeness we discuss the efficiency of the CGS factorization which can be performed on rectangular matrices, see for example Golub and Van Loan [5]. The largest p LCEs can thus be obtained by using the CGS factorization adapted for rectangular matrices. The method using CGS to obtain the largest p LCEs will be called CGSp.

3. Computational efficiency of HQRBp1 and HQRBp2

A QR decomposition and a matrix multiplication of a tangent map times the resulting Q are the two main operations performed at every iteration in the process of computing the LCEs. A count of the floating point operations (flops, as defined in [5]) for the QR decomposition and the product of the next tangent map and Q is given for HQRBp1, HQRBp2. For the sake of completeness, we also include CGSp in Table 1.

The values in Table 1 are for the computation of the largest p out of n LCEs, and it only shows terms in p and n which are of second order and higher. Depending on p and n one method may be computationally more efficient than another. For the extreme case of $p = 1$, all three methods have the coefficient $2n^2$ in the leading term. This is a considerable improvement when compared with the order n^3 term in the cost of computing all the LCEs, see [4], i.e. HQRBp1 and HQRBp2 are $\frac{5}{3}n$ times less expensive than HQRB. For $p = n$ (for large n), HQRBp1 is the most efficient with $\frac{10}{3}n^3$ flops which is the same as HQRB [4]; this is followed by CGSp with $4n^3$, and HQRBp2 with $5n^3$.

Fig. 1 shows the regions (for $3 \leqslant n \leqslant 30$ and $1 \leqslant p \leqslant n$) in which either HQRBp1 or HQRBp2 is more efficient. For small values of p, HQRBp2 is less

Table 1
Operation count for the factorization and action of multiplication by Q

Method	Multiplications/divisions	Additions/subtractions	Total
HQRBp1	$2n^2p - \frac{1}{3}p^3 - n^2$ $+\frac{1}{2}p^2 + 3np$	$2n^2p - \frac{1}{3}p^3 - n^2 + 2np$	$4n^2p - \frac{2}{3}p^3 - 2n^2$ $+5np + \frac{1}{2}p^2$
HQRBp2	$\frac{5}{2}np^2 + n^2p - \frac{5}{6}p^3$ $+\frac{1}{2}np + \frac{1}{2}p^2$	$2np^2 + n^2p - \frac{2}{3}p^3 - np$	$\frac{9}{2}np^2 + 2n^2p - \frac{3}{2}p^3$ $-\frac{1}{2}np + \frac{1}{2}p^2$
CGSp	$n^2p + np^2 + np$	$n^2p + np^2 + np$	$2n^2p + 2np^2 + 2np$

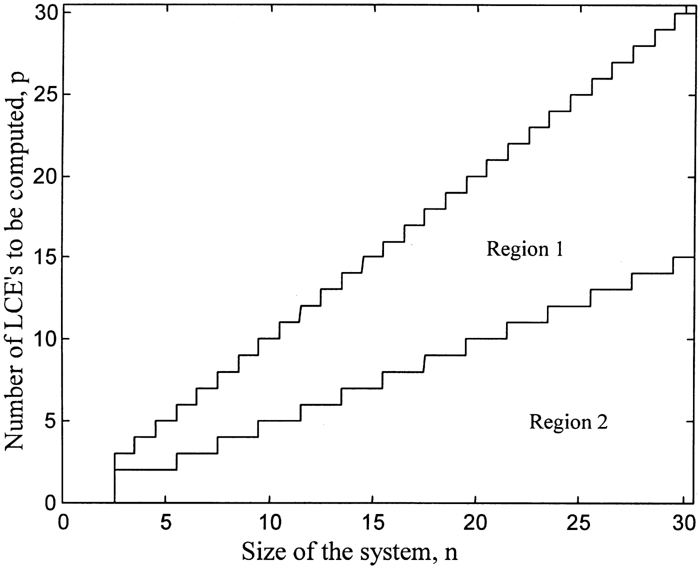

Fig. 1. Regions of efficiency: in Region 1, HQRBp1 is more efficient; in Region 2, HQRBp2 is more efficient.

expensive than HQRBp1. On the other hand for values of p close to n, HQRBp1 is more efficient than HQRBp2 (see Fig. 1). Asymptotically, (for large n) if less than about 48% of the LCEs are to be computed, HQRBp2 is numerically more efficient than HQRBp1. Furthermore, HQRB, which computes all the LCEs (see [4]), is more efficient than HQRBp2 if more than about 74% of the LCEs are to be computed.

A comparison of the three methods is shown on Fig. 2. Regions (for $3 \leqslant n \leqslant 30$ and $1 \leqslant p \leqslant n$) in which either of HQRBp1, HQRBp2 or CGSp is the most efficient are indicated. For p close to n, HQRBp1 is the most efficient; CGSp is the most efficient for the intermediate portion. If only the largest LCE ($p = 1$) is required to be computed, for $n > 6$ HQRBp2 is the most efficient, albeit by a small amount.

Fig. 3 shows a comparison of the flop count for the three methods when $n = 25$. A similar behavior is observed for smaller values of n, see insert in Fig. 3 for $n = 5$. This figure points out the *extent* to which one method is computationally more efficient than the others. We observe that HQPBp1 is significantly more efficient than HQPBp2 when p is close to n. For small p/n ratios, the difference in efficiency between HQRBp2 and HQRBp1 is not highly significant, though HQRBp2 is more efficient. Lastly, for any p/n ratio, CGS when compared to the more efficient of HQRBp1 and HQRBp2 is at most about 20% cheaper. The differences in efficiency being small, if any, the HQRB

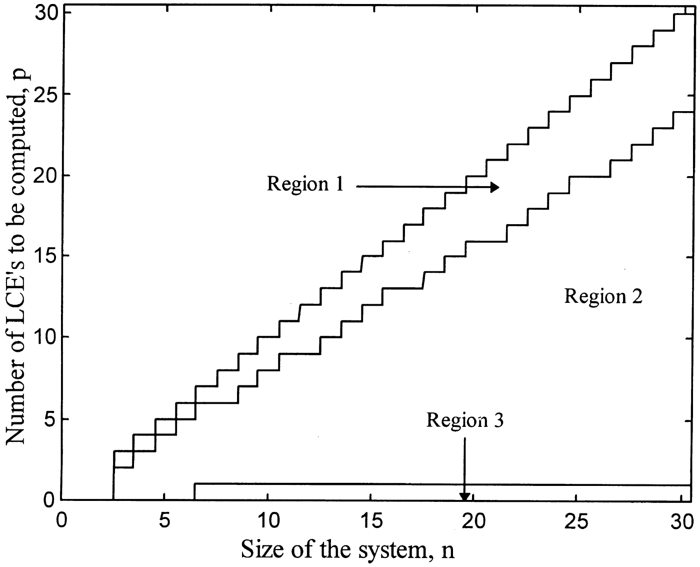

Fig. 2. Regions of efficiency: in Region 1, HQRBp1 is more efficient; in Region 2, CGSp is more efficient; and in Region 3, HQRBp2 is more efficient.

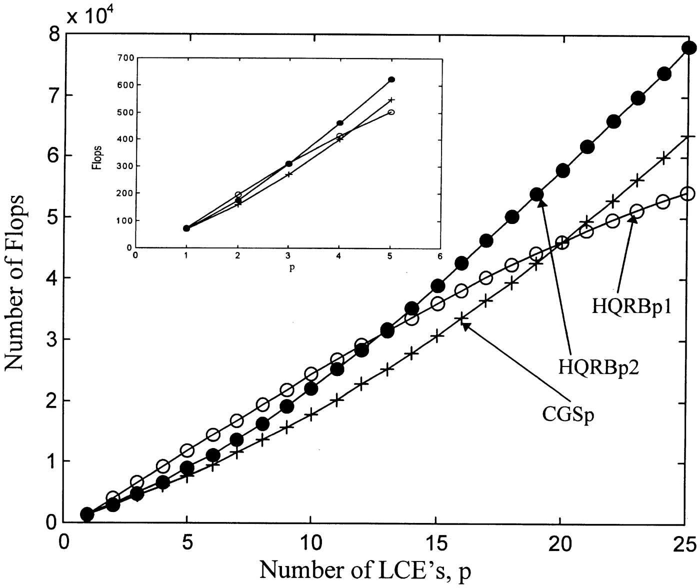

Fig. 3. Comparison of the efficiencies of the methods when computing p LCEs, for $n = 25$. See insert for $n = 5$. For $p = n$, the flop count for HBRBp1 is the same as that for HQRB [4].

methods developed in this paper thus appear more desirable, because of their superior stability, than CGSp.

4. Numerical results using HQRBp1 and HQRBp2

In modeling large dynamical systems, ring geometries are often used (see for example [7]). The largest LCEs are important indicators that help us understand the dynamics of these large systems. In the following example we consider two rings of oscillators which are coupled. Each of the two rings (Ring X and Ring Y) has n oscillators, and the two rings are connected through the coupling matrices C_{xy} and C_{yx}. See Fig. 4 for a drawing of the system.

The equations governing each of the oscillators are given by

$$\begin{aligned} x(i+1) &= Af(x(i),\boldsymbol{\alpha}) + C_{xy}f(y(i),\boldsymbol{\alpha}), \\ y(i+1) &= Bf(y(i),\boldsymbol{\beta}) + C_{yx}f(x(i),\boldsymbol{\alpha}), \end{aligned} \tag{6}$$

where

$$A = \begin{bmatrix} a & b & & & b \\ b & a & b & & \\ & b & a & b & \\ & & & \ddots & \\ b & & & b & a \end{bmatrix}, \qquad B = \begin{bmatrix} c & d & & & d \\ d & c & d & & \\ & d & c & d & \\ & & & \ddots & \\ d & & & d & c \end{bmatrix}, \tag{7}$$

and

$$\begin{aligned} f(x(i),\boldsymbol{\alpha}) &= \begin{bmatrix} 1-\alpha_1 x_1^2(i) & 1-\alpha_2 x_2^2(i) & 1-\alpha_3 x_3^2(i) & \cdots & 1-\alpha_n x_n^2(i) \end{bmatrix}^{\mathrm{T}}, \\ f(y(i),\boldsymbol{\beta}) &= \begin{bmatrix} 1-\beta_1 y_1^2(i) & 1-\beta_2 y_2^2(i) & 1-\beta_3 y_3^2(i) & \cdots & 1-\beta_n y_n^2(i) \end{bmatrix}^{\mathrm{T}}. \end{aligned} \tag{8}$$

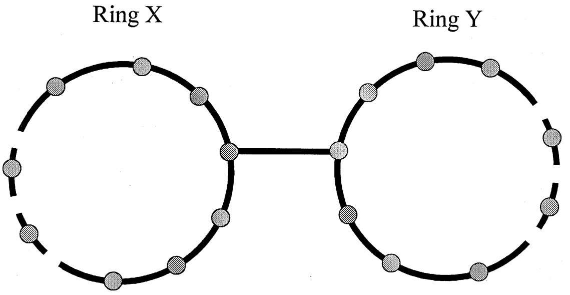

Fig. 4. Two coupled oscillator rings.

The matrices A, B, C_{xy} and C_{yx} are n by n, and the vectors $x(i+1)$, $y(i+1)$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are n by 1. With $n = 50$ and using the parameter values of $a = c = 0.75$, $b = d = 0.125$, $\beta_j = 1.2$ for all j, together with

$$\alpha_j = \begin{cases} 1.9 & \text{for } j = 10,\ 20,\ 30,\ 40,\ 50, \\ 1.2 & \text{otherwise,} \end{cases}$$

and

$$C_{xy}(i,j) = C_{yx}(i,j) = \begin{cases} 0.1 & \text{for } i = j = 20, \\ 0 & \text{otherwise,} \end{cases}$$

the following numerical results were obtained. To illustrate our algorithm the six largest LCEs are computed. For comparison purposes among the different methods it suffices to consider only 1,000 iterations. The computation of the LCEs was started after the first 10,000 trajectory points of the system had been computed. The six largest LCEs using the HQRBp2 after 1,000 iterations were 0.1722, 0.1559, 0.1457, 0.1261 and −0.0727 (note that the first five are positive).

A comparison of the computational efficiency between HQRB, CGS, CGSp, HQRBp1 and HQRBp2 is given in Table 2. The table shows the number of flops required to estimate the six largest (or all) LCEs in one iteration. Note that here $n = 50$ refers to the number of oscillators in a single ring, and that the complete system (the two coupled rings) have a total of 100 oscillators. The values presented in the table exclude the number of flops required to compute the trajectory and the tangent map. There is a clear advantage in using methods that just compute the six largest ($p = 6$) LCEs over methods that compute all of them. When comparing the methods for computing the six largest ($p = 6$) LCEs, the CGS is the more efficient one, followed closely (by about 5%) by HQRBp2. However, one must keep in mind that CGS (and thus CGSp) has been shown to present numerical instabilities which can lead to erroneous results [4].

An indicator of the computational efficiency in computing all the LCE's versus computing the largest p LCEs for a given method is the ratio of the flops needed to compute all the LCEs to the flops needed to compute the largest p LCEs. Table 3 shows the theoretical and the numerically computed values of the ratios of the flops per iteration between CGS and CGSp, and between HQRB and HQRBp2. For convenience, the theoretical ratios shown are for

Table 2
Flops per Iteration ($n = 50$, $p = 6$)

All LCEs		p LCEs ($p = 6$)		
HQRB	CGS	CGSp	HQRBp1	HQRBp2
336.8×10^4	402×10^4	12.8×10^4	22.3×10^4	13.5×10^4

Table 3
Flop ratios per iteration

Methods	Theoretical (leading term approximation)	Expected theoretical (leading term approx.)	Expected theoretical (exact formulas)	Actual computed
CGS/CGSp	$2(n/p)$	33.3	31.3	31.4
HQRB/HQRBp2	$1.67(n/p)$	27.7	24.8	24.9

large values of n and small values of p, using just the leading terms from Table 1. The expected theoretical values correspond to a system with 100 oscillators and when $p = 6$; the computed values correspond to the actual flop counts presented in Table 2. The fact that the ratio of the flops for CGS to CGSp is larger than the ratio of the flops for HQRB to HQRBp2 indicates that relative to the given method, more efficiency is gained when using CGSp than when using HQRBp2. The ratios obtained from the theoretical leading term approximation are in close agreement with the ratios obtained in actual practice. The ratios obtained from the exact expected theoretical formulas (formulas including all terms from Table 1) are extremely close to the ratios actually computed.

Another important aspect of using methods to compute just the largest p LCEs is the fact that there is a significant reduction in the storage requirements (depending on p and n) if the computed LCEs are required to be stored at every iteration.

5. Conclusions

Two efficient adaptations of the HQRB method (HQRBp1 and HQRBp2) for computing the largest p LCEs of an n dimensional discrete dynamical system are presented.

The HQRBp1 method is computationally the least expensive method when p is close to n. HQRBp2 is asymptotically more efficient than HQRBp1 for $p < 0.48n$, with the efficiency reversing when the inequality reverses. Both HQRBp1 and HQRBp2 are numerically backward stable since they are based on Householder transformations (see [4,6]).

References

[1] G. Benettin, L. Galgani, A. Giorgilli, J.M. Strelcyn, Lyapunov characteristic exponents for smooth dynamical systems and for hamiltonian systems a method for computing all of them, Part I and Part II, Meccanica 15 (1980) 9–30.

[2] K. Geist, U. Parlitz, W. Lauterborn, Comparison of different methods for computing Lyapunov exponents, Prog. Theor. Phys. 83 (5) (1990) 875–893.

[3] L. Dieci, E. Van Vleck, Computation of a few Lyapunov exponents for continuous and discrete dynamical systems, Appl. Numer. Math. 17 (3) (1995) 275–291.

[4] H. von Bremen, F. Udwadia, W. Proskurowski, An efficient QR based method for the computation of Lyapunov exponents, Physica D 101 (1997) 1–16.

[5] G. Golub, C. Van Loan, Matrix Computations, 2nd ed., Johns Hopkins University Press, Baltimore, 1993.

[6] J.H. Wilkinson, The Algebraic Eigenvalue Problem, Oxford University Press, Oxford, 1965.

[7] M.A. Matías, J. Güémez, V. Pérez-Muñuzuri, I.P. Mariño, M.N. Lorenzo, V. Pérez-Villar, Size instabilities in rings of chaotic synchronized systems, Europhysics Letters 37 (6) (1997) 379–384.