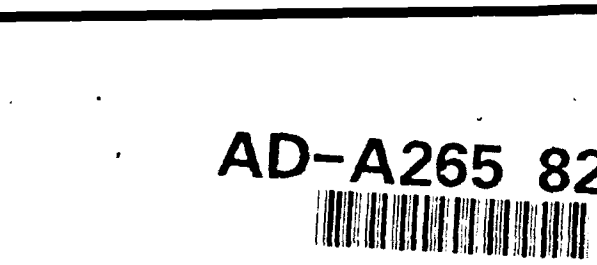


NPS-MA-93-015

NAVAL POSTGRADUATE SCHOOL

Monterey, California

CHAOTIC KEYSTREAM GENERATORS FOR ADDITIVE STREAM CIPHERS

by

Jeffery J. Leader

Technical Report For Period
January 1993 - April 1993

Prepared for: Naval Postgraduate School
Monterey, CA 93943

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943

Rear Admiral T.A. Mercer
Superintendent

Harrison Shull
Provost

This report was prepared in conjunction with research conducted for the Naval Postgraduate School and funded by the Naval Postgraduate School.

This report was prepared by:

Jeffery J. Leader
Assistant Professor of Mathematics

Reviewed by:

RICHARD FRANKE
Chairman

Released by:

PAUL J. MARTO
Dean of Research

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) NPS-MA-93-015	5 MONITORING ORGANIZATION REPORT NUMBERS NPS-MA-93-015
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) MA	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943	7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93943
8a NAME OF FUNDING/SPONSORING ORGANIZATION Naval Postgraduate School 8b OFFICE SYMBOL (If applicable) MA	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER OM,N
8c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification) Chaotic Keystream Generators for Additive Stream Ciphers	
12 PERSONAL AUTHOR(S) Jeffery J. Leader	
13a TYPE OF REPORT Technical 13b TIME COVERED FROM 1-93 TO 4-93	14 DATE OF REPORT (Year, Month, Day) 6 May 93 15 PAGE COUNT 13
16 SUPPLEMENTARY NOTATION	
17 COSATI CODES FIELD / GROUP / SUB-GROUP	18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Keystream generators, additive stream ciphers

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

We survey some current research concerning the generation of random and pseudorandom keystreams for use in cryptographic stream ciphers, including the use of chaotic discrete dynamical systems.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL Jeffery J. Leader	22b TELEPHONE (Include Area Code) 408-656-2335 22c OFFICE SYMBOL MA/Le

DD Form 1473, JUN 86 Previous editions are obsolete SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-013-6603

CHAOTIC KEYSTREAM GENERATORS FOR ADDITIVE STREAM CIPHERS

Jeffery J. Leader
Mathematics Department
Naval Postgraduate School
Monterey, CA 93943

April 1993

Abstract

We survey some current research concerning the generation of random and pseudorandom keystreams for use in cryptographic stream ciphers, including the use of chaotic discrete dynamical systems.

1 Introduction

Communications security is vital in the information age. For this reason, cryptography (the branch of cryptology dealing with the making of secure cipher systems; the other branch, cryptanalysis, attempts to break such systems) is no longer of concern only to the government, but also to private industry. Simple cipher systems such as those in [Sinkov] are easily broken today, if not by *ingenuity then by brute force computer attacks* [Bamford]; more sophisticated methods are therefore necessary. One cipher system used when security is essential (e. g., the "hot line" connecting the Pentagon to Moscow [Kahn]) is the one-time pad, or random Vernam cipher. Consider a message P to be encrypted; P is called the plaintext. We begin by converting P to binary form, say by using the ASCII code. The encoded plaintext is a binary string S of some length N. We now generate a bitstring R of length N, the entries of which are chosen uniformly and independently from $\{0, 1\}$; that is, R is a "random" bitstring (called the keystream). The enciphered message, or ciphertext (C), is formed as the bitwise exclusive-or of S and R,

$$C = S + R \pmod 2$$

where the sum is formed bit-by-bit. The ciphertext C may now be transmitted, and the receiving party may recover S as

$$\begin{aligned} S &= C - R \pmod 2 \\ &= C + R \pmod 2 \end{aligned}$$

(bitwise) since subtraction is the same as addition in binary arithmetic. Then S can be decoded to P using the original binary coding system.

This method is provably secure [Shannon] but has several drawbacks. First, generating truly random bits is a time-consuming and difficult process. Second, the one-time pad sequence R is as long as the (encoded form of the) message; this sequence must be known to both the sender and the receiver, creating serious logistical problems. Although this is done for sufficiently sensitive material [Bamford], for less sensitive material an additive stream cipher is often employed. Here the random bitstring R is replaced by a pseudorandom bitstring generated by some method (commonly a linear shift register; see section 3). In this case a small key of fixed length (the seed of the pseudorandom number generator) is all the information required to generate the enciphering sequence of length N. It might seem apparent that this system inherits the known security of the random Vernam cipher, but this depends on the quality of the pseudorandom number generator. We will discuss several measures of pseudorandomness in the next section.

Before doing so, however, we mention another way to shorten the keylength [Maurer 1]. Consider a large, publicly available list of truly random bits-a list much larger than any message which might be sent. Arrange it into K rows and T columns (so that there are a total of KT bits in the list). Once again we will form our ciphertext C from the encoded form S (of length N) of our plaintext P by

$$C = P + W \pmod 2$$

(bitwise), where W is a "random" bitstring chosen from the list as follows. We select a secret key, known only to the sender and the (intended) receiver. This key is a K-vector, with each entry an integer chosen from $\{1, \cdots, T\}$. For each of the K rows of the list of random bits, the corresponding element of the key specifies a column. Beginning with this column, we take the first N bits in that row and form a binary sequence w_i for $i = 1, \cdots, K$ (wrapping around to the first entry of the row if we reach its end). Then

$$W = w_1 + \cdots + w_n \pmod 2$$

(bitwise) is the desired keystream. This cipher system is strongly-randomized and requires a key of fixed length, but the logistics of generating and maintaining the public table may make this scheme impractical.

2 Pseudorandom Bitstreams

Now we consider the problem of generating a pseudorandom keystream R for use in an additive stream cipher. What properties should such a bitstring have? There is no universal agreement. However, it seems desirable that the bitstring be balanced, that is, have roughly the same proportion of ones and zeroes. If not, an enemy cryptanalyst could simply guess that the keystream was

$$R = \{1, 1, 1, \cdots, 1\}$$

(or, alternatively, all zeroes) and have a fair chance of reading at least part of the message. By extension, the sequence should probably have balanced runs, that is, all binary subsequences of length $n < N$ should appear equally often (roughly). For example, the keystream

$$R = \{0,1,0,1,\cdots,0,1\}$$

is balanced, but since the only runs of length two that appear are the subsequences $\{0,1\}$ and $\{1,0\}$, the cryptanalyst has an edge.

In addition, prediction of the sequence from a small part of it should be difficult (for someone without knowledge of the underlying generator, of course). The next bit test [Schrift] requires that prediction of the next bit in a sequence from the previous bits be not significantly better than 50% successful. A more common means of achieving the goal of unpredictability is to require that the autocorrelation of the proposed keystream be small for nonzero lags. In particular, the widely-used Golomb postulates state that a sequence is pseudorandom if it is balanced, has good runs properties, and has a small autocorrelation. What is meant by "good" and "small" depends on the application and the degree of security needed.

There are a variety of other measures of pseudorandomness in use, including the notions of per bit entropy, Maurer's universal statistical test [Maurer 2], and many more. In the next section we discuss one final test, the linear complexity profile.

3 Linear Shift Registers

One of the most common methods of generating pseudorandom sequences used in practice is the linear shift register. An n-stage linear shift register is the electronic implementation of an nth order recurrence relation of the form

$$r_i = a_1 r_{i-1} + a_2 r_{i-2} + \cdots + a_n r_{i-n} \tag{3.1}$$

where $a_1, \cdots, a_n$ are chosen from $\{0,1\}$, and the initial fill $\{r_0, \cdots, r_{n-1}\}$ (a binary sequence of length n, not consisting entirely of zeroes) is given. For example, the recurrence relation

$$r_i = r_{i-2} + r_{i-3} \tag{3.2}$$

can easily be implemented as a linear shift register [Heyman 1]. If the initial fill is $\{1,0,0\}$, i. e.

$$\begin{aligned} r_0 &= 1 \\ r_1 &= 0 \\ r_2 &= 0 \end{aligned}$$

then the successive iterates can be computed from the formula (3.2) as

$$\begin{aligned} r_3 &= r_1 + r_0 \\ &= 0 - 1 \end{aligned}$$

$$
\begin{aligned}
&= 1 \\
r_4 &= r_2 + r_1 \\
&= 0 + 0 \\
&= 0 \\
r_5 &= r_3 + r_2 \\
&= 1 + 0 \\
&= 1
\end{aligned}
$$

and so on. The sequence $\{r_0, r_1, r_2, r_3, \cdots\}$ generated by this method is

$$\{1, 0, 0, 1, 0, 1, 1, \cdots\} \tag{3.3}$$

where the seven bits indicated repeat periodically. Although this may seem undesirable at first, it is the best possible result. To see this, note that in the formula (3.2) each new iterate is determined by the three preceding iterates, and hence the sequence must repeat as soon as a three-tuple is repeated. As there are only seven possible nonzero three-tuples in this case (the three-tuple $\{0, 0, 0\}$ would lead to an infinite sequence of zeroes, which is not very pseudorandom), and all are present in the sequence (3.3) (when repeated periodically), the sequence must begin to repeat itself. More generally, an nth order recurrence relation of the form (3.1) leads to a binary sequence which is periodic with period at most

$$2^n - 1$$

and possibly less. By associating with the formula (3.1) a characteristic polynomial (in the usual way), it is possible to characterize those linear shift registers that give rise to sequences of maximal period (called m-sequences). These full length shift registers are the ones employed in practice. In addition to having maximal period, it follows that in an m-sequence all n-tuples appear exactly once (except for the n-tuple of all zeroes).

Now suppose that one is confronted with a binary sequence of length N which is known to have been generated by an n-stage linear shift register. It would appear from (3.1) that any subsequence of $2n$ consecutive bits

$$\{b_0, \cdots, b_{2n-1}\}$$

could be used to set up a linear system in the unknown coefficients $\{a_1, \cdots, a_n\}$

$$
\begin{aligned}
b_n &= a_1 b_0 + a_2 b_1 + \cdots + a_n b_{n-1} \\
b_{n+1} &= a_1 b_1 + a_2 b_2 + \cdots + a_n b_n \\
&\vdots \\
b_{2n-1} &= a_1 b_{n-1} + a_2 b_n + \cdots + a_n b_{2n-2}
\end{aligned}
$$

which might then be solved, giving the recurrence relation. In practice, the Berlekamp-Massey [Massey] algorithm, or a variant, is used. This method does not attempt to solve the system given above but instead finds the coefficients in a recurrence relation that *could*

have been used to generate the sequence; if the sequence in question is not an m-sequence, the recurrence relation generated by the Berlekamp-Massey algorithm may be of lower order than the one actually used to generate the sequence in the first place. The order of the recurrence relation found by this algorithm is called the linear complexity of the sequence, and a higher linear complexity generally indicates a greater degree of pseudorandomness.

Although m-sequences have many interesting and desirable properties (not all of which can be mentioned here), including good autocorrelation properties, we mention some drawbacks. First, notice that an n-stage linear shift register produces a sequence whose period is bounded in terms of n. Second, the sequence can be completely simulated from the knowledge of only $2n$ bits. Of course, these $2n$ pseudorandom bits are masked in the ciphertext by the message, just as the message is masked by the keystream, and so finding such a string of bits generally requires some extra information (knowledge of a small part of the plaintext, for example). Finally, consider the sequence (of length N)

$$\{0,0,0,\cdots,0,1\} \tag{3.4}$$

i.e. all zeroes except for the last bit. Since this is an N-tuple, but not a p-tuple for $p < N$, the linear complexity of this sequence is N. Hence a very "bad" sequence can have arbitrarily high linear complexity (m-sequences do not possess this problem).

One means of detecting sequences such as (3.4) is the linear complexity profile [Rueppel]. Given a binary sequence of length N, we plot the linear complexity of the leading subsequence of length i, L_i, versus $i (i = 1, \cdots, N)$. The sequence $\{L_1, \cdots, L_N\}$ is a nondecreasing sequence of nonnegative integers. For a "good" sequence, the leading subsequences should have linear complexity near $i/2$ (since the first i points could be used to determine a recurrence relation of order $i/2$). For the sequence (3.4), the linear complexity profile would be flat until the last element ($L_i = 0, i = 1, \cdots N-1$), which would then jump up to the linear complexity of the sequence ($L_N = N$). This does not follow the line with slope $1/2$ and hence would not be considered a "good" sequence.

Related to the linear complexity profile is the jump complexity profile [Niederreiter]. For $i = 1, \cdots, N$, we form the first difference of $\{L_1, \cdots, L_i\}$, and define the jump complexity profile P_i to be the number of positive integers (the "jumps") in the vector of differences. It is immediate that $P_i \leq L_i$, and for a "good" sequence it can be shown that P_i should follow the line with slope $1/4$ when plotted versus $i (i = 1, \cdots, N)$.

4 Cryptographically Strong Sequences

Are m-sequence pseudorandom? In a sense, they offer almost too much regularity to appear random. However, they seem to have all the properties we might desire in a keystream–balance, balanced runs, good autocorrelation, optimal linear complexity with a good linear complexity profile, and period length that can be made as large as desired. In this sense they are quasirandom, that is, random enough for our purposes. Of course, all we really are interested in is a sequence which will give a cipher that is as difficult to break as possible–pseudorandomness is merely the most obvious method of finding such sequences. In a cryptographic setting, we generally speak of a keystream as being cryptographically

strong if it would make a "good" keystream, in a sense that is rarely made precise; generally, pseudorandomness is a major aspect of cryptographic strength.

There is another pragmatic consideration in choosing a keystream: errors. Because of the possibility of transmission errors, the encoding used to transfosrm the plaintext to its binary form S often includes some redundancy so that the code can be error-correcting. For sufficiently short messages, the entire plaintext may be encoded an odd number of times and, in the event of discrepancies between the various copies of the message, the majority rules. Other more sophisticsated methods are often used.

However, keystream errors can also occur. For this reason (among others), the notion of k-complexity has been proposed by [Stamp]. Since the term k-complexity is also used in another context in cryptography (namely, in describing certain nonlinear shift registers), we will use the term linear komplexity (of order k) instead. For a given sequence R, the komplexity of order k is defined as the smallest linear complexity of all sequences which differ from R in at most k places, which can be interpreted as the worst case result of at most k errors occurring in the keystream sequence. For example, sequence (3.4) has linear komplexity (or complexity of order zero) N, but the komplexity of order one is clearly zero, since changing a single bit (the final one) gives the sequence

$$\{0, 0, 0, \cdots, 0\}$$

which, by convention, has linear complexity zero (the lowest possible). Hence this keystream is not safe with respect to transmission errors. Another interpretation is to say that the komplexity of order one has detected the weaknesses that would have been seen in the linear complexity profile.

More generally, a linear komplexity profile can be created. The example (of length N)

$$\{1, 1, 1, \cdots, 1\}$$

(which has unit linear complexity but can be made into a sequence of zero linear complexity with exactly N changes) shows that the komplexity of order k should be computed for $k = 0, \cdots, N$ (unless it should reach zero before then). However, at most $N/2$ changes can always give a sequence of all ones or all zeroes (by changing whichever there are fewer of in the sequence—ones or zeroes), and so the height of the profile is at most unity for values above $N/2$, and therefore not especially interesting. The linear komplexity profile may be viewed in two ways. First, a sharp drop for small k indicates a cryptographically weak sequence (as a relatively small shift register will approximate the sequence in all but k places). Second, if the number of potential transmission errors can be estimated, then the resulting worst case linear complexity can be checked for acceptability.

5 Chaotic Keystream Generation Schemes

Linear shift registers are commonly used for the generation of keystream sequences (generally with their output further scrambled by a nonlinear Boolean function before being used in the cipher). However, their fixed maximal period (for a given number of stages) and the fact

that they can be perfectly simulated from a relatively small number of bits of pure keystream (2.N for an N-stage register, using the Berlekamp-Massey algorithm), are causes for concern. For this reason it is important to investigate other methods of running keystream generation. An obvious choice for such a generator is a chaotic recurrence relation. One investigation of such a scheme is reported in [Forré], who used the Hénon map [Hénon]

$$\begin{aligned} x_{n+1} &= 1 - 1.4x_n^2 + y_n \\ y_{n+1} &= 3y_n \end{aligned}$$

(where (x_0, y_0) may be chosen anywhere in a certain quadrilateral which includes the origin) as the basis of such a scheme. Her approach was to consider the strange attractor (Fig. 1) generated by the map (independent of the initial condition (x_0, y_0)) and to estimate the median of the abscissas of the orbits of the Hénon map on this attractor (note that this is different from the median of the x-values of the attractor). A more accurate investigation of this scheme was carried out in [Heyman 1], where it was determined experimentally that the median x-value of an orbit on the attractor is

$$\dot{x} \doteq .4098$$

independent of the initial condition. We now choose an initial condition and iterate the Hénon map, giving the orbit

$$(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots$$

and we generate a corresponding binary sequence by using the median value as a split point (Fig. 2), i. e.

$$R_i = \begin{cases} 1 & \text{if } x_i \geq \dot{x} \\ 0 & \text{otherwise} \end{cases} \qquad (5.1)$$

giving a binary sequence R, called a binary Hénon sequence. Heyman has shown (experimentally) that such sequences have extremely long periods (as might be expected from a chaotic map), are balanced, and have good autocorrelation properties and linear complexity profile. However, as noted also by Forré, the runs are poorly balanced: subsequences containing primarily alternating patterns of zeroes and ones tend to occur noticeably more frequently than those containing longer runs of all zeroes or all ones (although runs of consecutive ones as long as 23 have been found by this author). In fact, [Fontana] has shown that the four-tuple

$$\{1, 1, 0, 0\}$$

can never occur with this split point (and eight of the thirty-two possible five-tuples are unrealizable, including the five-tuple of all zeroes). Despite the attractiveness of the high periods, this is enough to rule out Hénon generated keystreams. Preliminary work indicates that using a single split as in (5.1) on other two-dimensional maps possessing strange attractors leads to similarly decimated n-tuple profiles, even for relatively small values of n [Leader]. This is unacceptable for cryptographic purposes; however, since a chaotic map must be homeomorphic to the shift map on two symbols [Devaney], there must be a way of converting

the dynamical sequence into a binary sequence—namely, the topological conjugacy—such that perfect balancing of the runs is obtained. Evidently (5.1) is not such a homeomorphism, despite its suggestive similarity to the topological conjugacy used to conjugate the quadratic map to code space [Devaney].

The logistic equation, a chaotic map, was one of the first computer pseudorandom number generators ever used [Peitgen], and research continues on the possible use of chaotic dynamical systems as pseudorandom bit generators. Other nonstandard (and nonchaotic) methods of pseudorandom number generation are being investigated as well [Nisan].

6 Conclusions

In cryptography as well as in other areas (such as simulation), there is a need to produce large numbers of pseudorandom numbers quickly and without certain statistical defects. The particular criteria to be met depend on the application but, in cryptography, generally include balance, balanced runs, good autocorrelation properties, and a good linear complexity profile; a large period is also desirable. Linear shift registers are the most commonly used method (generally in conjunction with a nonlinear Boolean function to further mask the output), but they have certain drawbacks. Chaotic discrete dynamical systems appear to offer an interesting alternative, but more work is necessary in this area. Initial investigations indicate potential problems (the balancing of the n-tuples) but also potential payoffs (the high periods and the nonlinearity, a defense against linear attacks). In [Heyman 2] evidence is presented that a simple artificial neural network can identify the type of keystream used (linear shift register vs. Hénon vs. linear congruential pseudorandom number generators), as well as simulate a linear shift register (in the fashion of Berlekamp-Massey, although using more bits) and effectively predict the next Hénon bit at about 70% accuracy, violating the next-bit test. This research highlights additional weaknessess of both the linear and nonlinear methods; however, the fact that the neural net completely simulates (i. e. gives perfect prediction for) the shift register but is imperfect for the Hénon method shows another advantage of the chaotic keystream generator (at least for this type of attack).

Bibliography

[Bamford] James Bamford, *The Puzzle Palace: A Report on America's Most Secret Agency*. Penguin Books 1982

[Devaney] Robert L. Devaney, *An Introduction to Chaotic Dynamical Systems* (2nd ed.). Addison-Wesley 1989

[Fontana] Antonio Fontana, "The Forré-Heyman Symbolic Dynamics of the Hénon Map". NPS Masters Thesis, expected June 1993

[Forré] Réjane Forré, The Hénon Mapping as a Binary Keystream Generator, preprint

[Hénon] Michel Hénon, A Two-Dimensional Mapping with a Strange Attractor. Comm. Math. Physics 50 pp. 69-77, 1976

[Heyman 1] James E. Heyman, "On the Use of Chaotic Dynamical Systems to Generate Pseudorandom Bitstreams", NPS Masters Thesis, March 1993

[Heyman 2] James E. Heyman and Jeffery J. Leader, Neural Network Identification of Keystream Generators, submitted as an NPS Technical Report

[Kahn] D. Kahn *The Codebreakers: The Story of Secret Writing*, MacMillan Co., New York 1967

[Leader] Jeffery J. Leader, to be submitted

[Massey] J. L. Massey, "Shift-register Synthesis and BCH decoding", *IEEE Transactions on Information Theory*, Vol. IT-15, No. 1, pp. 122-127, January 1969

[Maurer 1] Ueli M. Maurer, "A provably-secure strongly-randomized cipher", in I. B. Damgård (ed.), Advances in Cryptology - EUROCRYPT'90, Springer-Verlag 1991

[Maurer 2] Ueli M. Maurer, "A Universal Statistical Test for Random Bit Generators", pp. 409-420 in A. J. Menezes and S. A. Vanstone (eds.), Advances in Cryptology - CRYPTO'90, Springer-Verlag 1991

[Niederreiter] Harald Niederreiter, "The Linear Complexity Profile and the Jump Complexity of Keystream Sequences", in I. B. Damgård (ed.), Advances in Cryptology - EUROCRYPT'90, Springer-Verlag 1991

[Nisan] Noam Nisan, *Using Hard Problems to Create Pseudorandom Number Generators*, MIT Press 1992

[Peitgen] Heinz-Otto Peitgen, Hartmut Jürgens and Dietmar Saupe, *Chaos and Fractals: New Frontiers of Science*, Springer-Verlag 1992

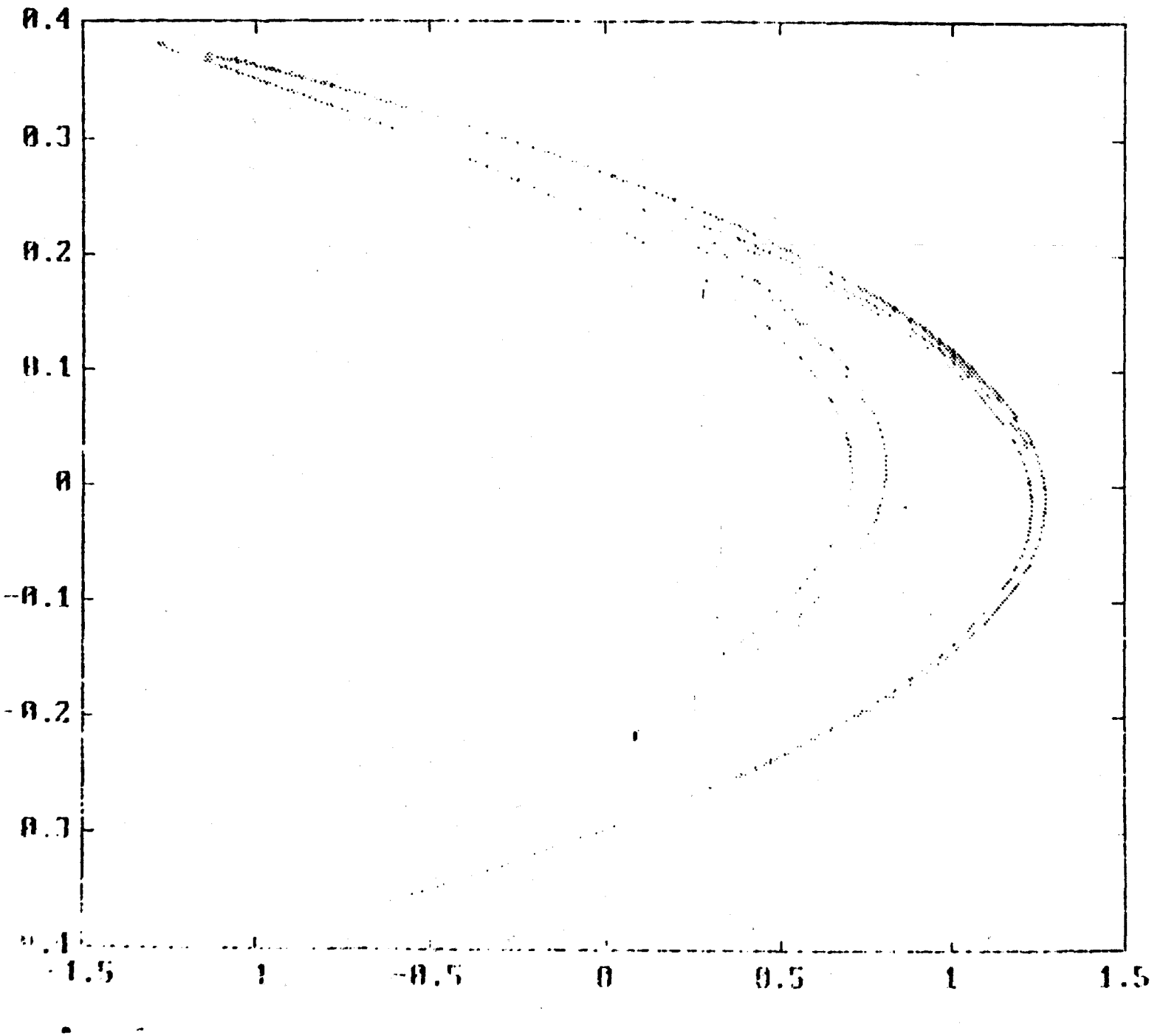
[Rueppel] R. A. Rueppel, *Analysis and Design of Stream Ciphers*, Springer-Verlag 1986

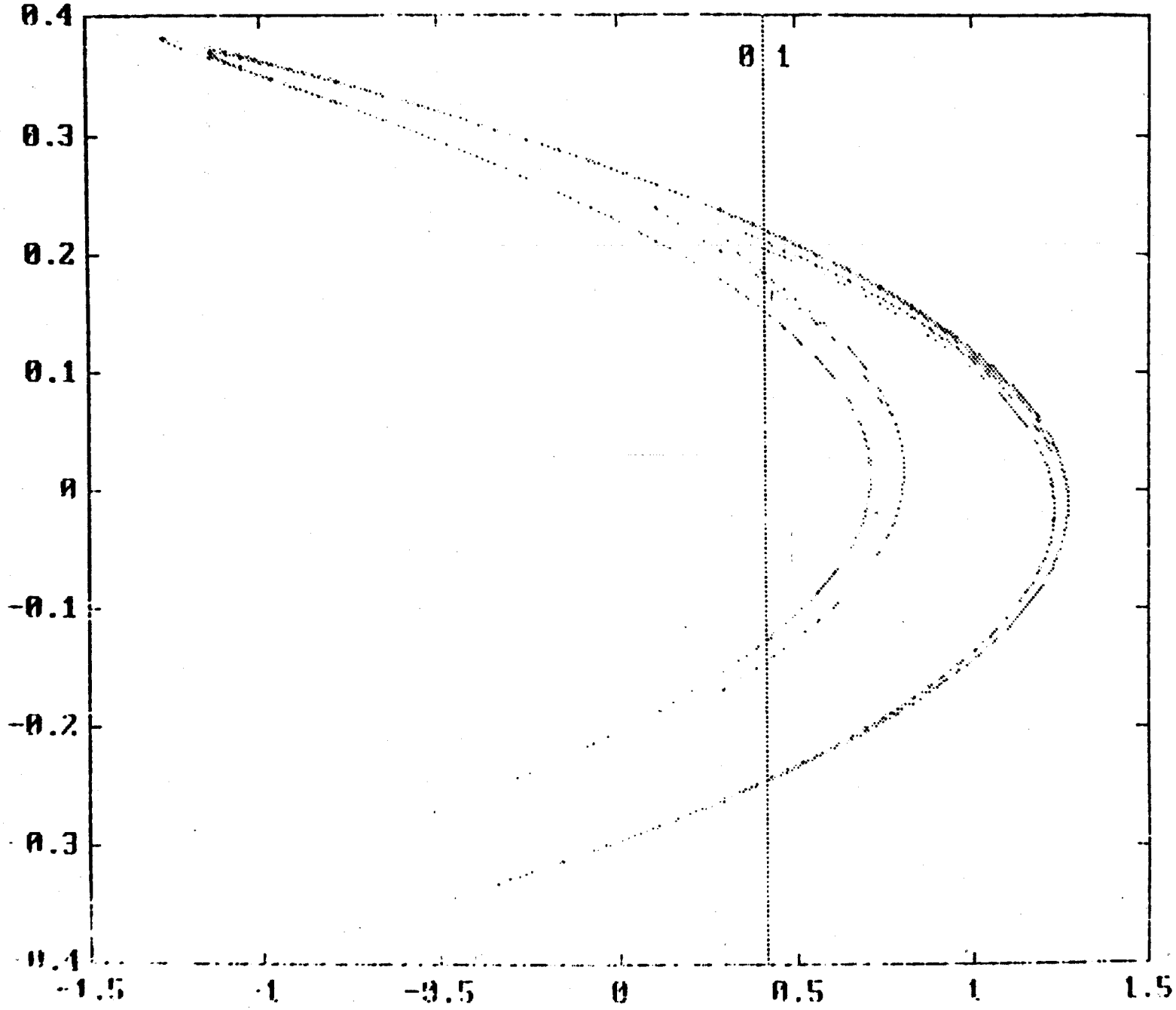
[Schrift] A. W. Schrift and A. Shamir, "On the Universality of the Next Bit Test", pp. 394-408 in A. J Menezes and S. A. Vanstone (eds.), Advances in Cryptology - CRYPTO'90, Springer-Verlag 1991

[Shannon] Claude E. Shannon, "Communication theory of secrecy systems", *Bell System Technical J.*, Vol. 28 pp. 656-715, 1949

[Sinkov] Abraham Sinkov, *Elementary Cryptanalysis: A Mathematical Approach*, Mathematical Association of America 1966

[Stamp] Mark Stamp, *A Generalized Linear Complexity*, Ph.D. Dissertation, Texas Tech University, Lubbock, TX 1992


0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-1.5
-1
-0.5
0
0.5
1
1.5


0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
0 1
-1.5
-1
-0.5
0
0.5
1
1.5

DISTRIBUTION LIST

Director (2)
Defense Tech Information Center
Cameron Station
Alexandria, VA 22314

Research Office (1)
Code 81
Naval Postgraduate School
Monterey, CA 93943

Library (2)
Code 52
Naval Postgraduate School
Monterey, CA 93943

Professor Richard Franke (1)
Department of Mathematics
Naval Postgraduate School
Monterey, CA 93943

Dr. Jeffery J. Leader, Code MA/Le (10)
Department of Mathematics
Naval Postgraduate School
Monterey, CA 93943

LT James Heyman (1)
PSC 825 Box 58
FPO AE 09627

Dr. Hal Fredricksen, Code MA/Fs (1)
Department of Mathematics
Naval Postgraduate School
Monterey, CA 93943

Dr. Ismor Fischer, Code MA/Fi (1)
Department of Mathematics
Naval Postgraduate School
Monterey, CA 93943

Dr. Todd A. Rovelli (1)
Division of Applied Mathematics
Brown University
Providence, RI 02912

LT Antonio Fontana, Code MA (1)
Naval Postgraduate School
Monterey, CA 93943

Dr. Herschel H. Loomis, Jr., Code EC (1)
Naval Postgraduate School
Monterey, CA 93943