$N(n, w, 2u)$. Then, whenever

$$M(n, w, 2u) = M(n-1, w-1, 2u) + M(n-1, w, 2u),$$

we must also have

$$(n-w)M(n-1, w-1, 2u) = wM(n-1, w, 2u). \quad (14)$$

If (14) is not satisfied in these cases, lower $M(n, w, 2u)$ by one. A final useful, though simple, relationship for establishing realizability is given by

$$N(n_1 + n_2, w_1 + w_2, 2u_1 + 2u_2) = \min \{N(n_1, w_1, 2u_1), N(n_2, w_2, 2u_2)\}. \quad (15)$$

C. V. Freiman
IBM Corp.
Thomas J. Watson Research Ctr.
Yorktown Heights, N. Y.

Application of Lyapunov's Direct Method to the Error-Propagation Effect in Convolutional Codes

Introduction

The convolutional type of code, discovered by P. Elias,[1] has assumed a central role in coding theory owing to its use in several practical decoding schemes.[2-6] Encoding is performed by a linear digital filter resulting in a continuous and queue-free encoding operation. One feature of the decoding process, however, has resulted in certain misgivings, namely the tendency of a decoding error to trigger a succession of further decoding errors. Certain strategies, such as periodic resynchronization,[7] have been suggested to control this error-propagation effect, but at the expense of introducing encoding queues. An alternative to such artificial means of limiting error propagation is the possibility that the decoder will itself "reconverge" to correct operation after a short burst of erroneous decoding decisions. That possibility is the subject of this communication. It is shown below that the error-propagation effect is closely related to the stability of a binary nonlinear-feedback shift register. This stability problem is analyzed with the aid of a modified form of Lyapunov's direct method. As an example, a practical convolutional decoder is analyzed by this method and it is shown that automatic reconvergence is obtained.

Formulation of the Problem

For ease of presentation, only binary, rate one-half, convolutional codes will be considered. In this case, there is a single sequence of information symbols which can be represented in delay-operator notation as

$$I(D) = i_0 \oplus i_1 D \oplus i_2 D^2 \oplus \cdots ,$$

where i_j is the information symbol input to the encoder at time unit j. The encoder forms a parity sequence which is given by

$$P(D) = p_0 \oplus p_1 D \oplus p_2 D^2 \oplus \cdots = I(D)G(D),$$

where

$$G(D) = g_0 \oplus g_1 D \oplus g_2 D^2 \oplus \cdots \oplus g_m D^m$$

is the code-generating polynomial whose choice fixes the code. p_j is the parity symbol formed at time j by the encoder. Both $I(D)$ and $P(D)$ are transmitted. The decoder input differs from $I(D)$ and $P(D)$ by the addition of the information and parity noise sequences, $E^i(D)$ and $E^p(D)$, respectively, where

$$E^i(D) = e_0^i \oplus e_1^i D \oplus e_2^i D^2 \oplus \cdots$$

and

$$E^p(D) = e_0^p \oplus e_1^p D \oplus e_2^p D^2 \oplus \cdots ,$$

and where $e = 1$ indicates that the corresponding symbol was received in error.

The decoder first forms the syndrome, or parity check, sequence $S(D)$ by encoding the received information symbols and adding the parity symbols thus formed to the received parity symbols. It readily follows that

$$S(D) = E^i(D)G(D) \oplus E^p(D) = s_0 \oplus s_1 D \oplus s_2 D^2 \oplus \cdots$$

or

$$S(D) = (e_0^i \oplus e_1^i D \oplus e_2^i D^2 \oplus \cdots)G(D) \oplus e_0^p \oplus e_1^p D \oplus e_2^p D^2 \oplus \cdots . \quad (1)$$

The decoding algorithm is the rule for deciding upon the value of e_0^i (*i.e.*, deciding whether or not the first information symbol was received correctly) from the first $m + 1$ terms of $S(D)$. Let $e_0^{i\Delta}$ denote the decoding decision for e_0^i. The decoder then prepares to determine e_1^i by first removing the effect of its previous decision, *i.e.*, by adding $e_0^{i\Delta}G(D)$ to the syndrome sequence. The modified syndrome sequence that results, excluding the time unit zero terms which are discarded, is given by

$$D[(e_0^i \oplus e_0^{i\Delta})(g_1 \oplus g_2 D \oplus \cdots \oplus g_m D^{m-1}) \oplus (e_1^i \oplus e_2^i D \oplus \cdots)G(D) \oplus e_1^p \oplus e_2^p D + \cdots],$$

from which it is clear how the effect of e_0^i is removed by a correct decoding decision. Moreover, by comparison to (1) it can be seen that decoding may proceed sequentially using the first $m + 1$ terms of the altered syndrome sequence to determine $e_1^{i\Delta}$ according to the same algorithm used to find $e_0^{i\Delta}$. The obvious difficulty occurs when $e_0^i \neq e_0^{i\Delta}$. The altered syndrome sequence then differs from its proper value by the spurious addition of $D(g_1 \oplus g_2 D \oplus \ldots \oplus g_m D^{m-1})$, and hence it is more likely that subsequent decoding decisions will be incorrect. Thus a decoding error tends to propagate.

The study of the error-propagation phenomenon will now be reduced to a "stability" analysis of a nonlinear-feedback shift register (NFSR, for short). The relevant portion of a decoder is shown in Fig. 1 and is seen to constitute a NFSR. The first m terms of the syndrome sequence are stored in the shift register, and the current input is s_m, at the time when the decoder forms $e_0^{i\Delta}$. Let the vector $\mathbf{s} = (s_0, s_1, \ldots s_{m-1})$ represent the shift-register contents and let $\mathbf{0}$ denote the all-zero vector. $\mathbf{s}$ will be referred to as the state vector, or simply state, of the NFSR. The decoding algorithm

Manuscript received July 16, 1963.

[1] P. Elias, "Coding for noisy channels," 1955 IRE Convention Record, pt. 4, pp. 37–46.

[2] J. Wozencraft and B. Reiffen, "Sequential Decoding," The Technology Press of M.I.T., Cambridge, Mass., and John Wiley and Sons, Inc., New York, N. Y.; 1961.

[3] D. Hagelbarger, "Recurrent codes: easily mechanized, burst-correcting, binary codes," *Bell Sys. Tech. J.*, vol. 38, pp. 969–984; July, 1959.

[4] J. Ziv, "Successive decoding scheme for memoryless channels," IEEE Trans. on Information Theory, vol. IT-9, pp. 97–104; April, 1963.

[5] R. Fano, "A Hueristic discussion of probabilistic decoding," IEEE Trans. on Information Theory, vol. IT-9, pp. 64–74; April, 1963.

[6] J. Massey, "Threshold Decoding," The Technology Press of M. I. T., Cambridge, Mass.; 1963.

[7] J. Wozencraft and B. Reiffen, op. cit., p. 67.

is represented by the function $F(s_m, \mathbf{s})$, that is, $F(s_m, \mathbf{s}) = e_0^{i\Delta}$. For any reasonable decoding algorithm, $F(0, \mathbf{0}) = 0$ since this is the case where all parity checks are satisfied.

From Fig. 1 it should be clear that m consecutive correct decoding decisions will clear the decoder of any spurious symbols introduced by a decoding error and hence will terminate the error propagation. The ability of the decoder to effect such a "reconvergence" is conveniently studied by considering the shift register to be loaded with some initial state $\mathbf{s}$ and the syndrome input sequence to be all zeroes, *i.e.*, all succeeding parity checks are satisfied. The shift register will enter state $\mathbf{0}$ when and only when reconvergence has been achieved. Thus the problem of studying error propagation reduces to the study of the autonomous behavior of the NFSR shown in Fig. 1.

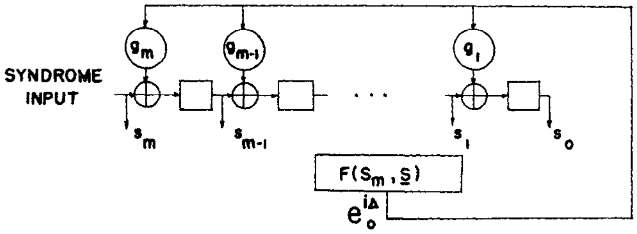

Fig. 1—Decoder NFSR.

Direct Method of Lyapunov

Considering only its autonomous behavior, *i.e.*, $s_m = 0$, the NFSR of Fig. 1 is represented by a difference equation of the form

$$\theta \mathbf{s} = \mathbf{f}(\mathbf{s}), \tag{2}$$

where $\mathbf{f}$ is a vector function and θ is the next-state operator. $\theta\mathbf{s}$ denotes the new state after one shift of the NFSR, $\theta^N\mathbf{s}$ denotes the new state after N shifts. Since $F(0, \mathbf{0}) = 0$, it is easy to see from Fig. 1 that $\theta\mathbf{0} = \mathbf{0}$ and hence that $\mathbf{0}$ is an *equilibrium state* of the NFSR. This is motivation for the following definition:

Definition 1

The NFSR is *stable* if, for every state $\mathbf{s}$, there exists some N such that $\theta^N\mathbf{s} = \mathbf{0}$. It is *unstable* if it is not stable.

In other words, if the NFSR is stable, it will reconverge eventually after a decoding error.

Stability analysis of systems whose variables are real numbers is facilitated by the direct method of Lyapunov.[8] The key concept in this method is the use of a Lyapunov function through which stability or instability may be established without full knowledge of the solution of the equations of the system. This method will now be modified to study the stability of the NFSR of Fig. 1, which is a finite-state discrete-time system.

Consider the binary scalar function $V(\mathbf{s})$ with the properties that $V(\mathbf{s}) = 0$ for $\mathbf{s}$ in set A and $V(\mathbf{s}) = 1$ for $\mathbf{s}$ in $\bar{A}$ where A is any set and $\bar{A}$ denotes the complement of A. Let $\Delta V(\mathbf{s})$ be defined by

$$\Delta V(\mathbf{s}) = V(\theta\mathbf{s}) \oplus V(\mathbf{s}).$$

Then ΔV gives the change of V when a state $\mathbf{s}$ is shifted in the NFSR. A state is shifted from A to $\bar{A}$ or from $\bar{A}$ to A if and only if $\Delta V = 1$ for that state. Hence, $\Delta V = 0$ for $\mathbf{s} = \mathbf{0}$ always. The following theorem is immediate:

Theorem 1

Let $V_0(\mathbf{s})$ have the property that $V_0(\mathbf{0}) = 0$ and $V_0(\mathbf{s}) = 1$ for $\mathbf{s} \neq \mathbf{0}$. Then the NFSR is unstable if $\Delta V_0 = 0$ for all $\mathbf{s}$ and is stable if $\Delta V_0 = 1$ for all $\mathbf{s} \neq \mathbf{0}$.

This theorem corresponds to the asymptotic stability theorem of Lyapunov. However, the conditions are too strong to be useful in analyzing a practical decoder. Stability can be established from Theorem 1 only when all nonzero states jump to $\mathbf{0}$ in a single shift, and instability can be established from Theorem 1 only when no nonzero states jump to $\mathbf{0}$. In the practical case, some states jump to $\mathbf{0}$ in one shift and some do not, *i.e.*, $\Delta V_0 = 1$ for some nonzero states and $\Delta V_0 = 0$ for others. This corresponds to the ambiguous case for real-number systems. For the binary system, the information about stability can still be obtained by following the steps outlined here:

Let A_0 be the set containing $\mathbf{0}$ alone. Form $V_0(\mathbf{s})$ as in Theorem 1. Let S_1 be the set of all $\mathbf{s}$ such that $\Delta V_0(\mathbf{s}) = 1$ and let $A_1 = A_0 \cup S_1$. The states in S_1 jump to $\mathbf{0}$ in one shift. If $\bar{A}_1 = \Phi$, where Φ is the empty set, or if $S_1 = \Phi$, then stability or instability, respectively, is established from Theorem 1. Otherwise, form $V_1(\mathbf{s})$ such that $V_1(\mathbf{s}) = 0$ for $\mathbf{s}$ in A_1 and $V_1(\mathbf{s}) = 1$ for $\mathbf{s}$ in $\bar{A}_1$. Let S_2 be the set of all $\mathbf{s}$ such that $\Delta V_1(\mathbf{s}) = 1$ and let $A_2 = A_1 \cup S_2$. The states in S_2 jump to $\mathbf{0}$ in two shifts. The process is repeated a finite number of times M (always $M \leq 2^m$) until $\bar{A}_M = \Phi$ or $S_M = \Phi$. In the former case, all states go to $\mathbf{0}$ in M or fewer shifts. In the latter case, there exist states which never go to $\mathbf{0}$. The following theorem is obtained:

Theorem 2

Repeating the steps outlined above until, in a finite number of steps M, $\bar{A}_M = \Phi$ or $S_M = \Phi$, then the NFSR is stable in the first case and is unstable in the second case.

The functions V_i are easily obtained. Let

$$V_0(\mathbf{s}) = 1 \oplus \prod_i (s_i \oplus 1),$$

where the s_i are the components of $\mathbf{s}$, and let

$$L_j(\mathbf{s}) = \sum_{\mathbf{s}' \text{ in } S_j} \prod_i (s_i \oplus s_i' \oplus 1).$$

Then the functions

$$V_n(\mathbf{s}) = V_0(\mathbf{s}) \oplus \sum_{j=1}^{n} L_j \qquad n = 1, 2, \cdots M$$

meet the conditions outlined above. Computation is facilitated by noting that

$$\Delta V_n(\mathbf{s}) = L_n(\theta\mathbf{s}).$$

An example will now be given to illustrate this process:

Example: The NFSR in Fig. 2 is the syndrome portion of a practical double-error-correcting decoder for a rate 1/2 convolutional code having $g_0 = g_3 = g_4 = g_5 = 1$.[9] $F(s_5, \mathbf{s})$, in this case, is a threshold function which takes on value 1 whenever three or more of the inputs $(s_0, s_3, s_4, s_1 \oplus s_5)$ are equal to one. Let the state $\mathbf{s}$ be represented by the decimal integer $s_0 + 2s_1 + 2^2s_2 + 2^3s_3 + 2^4s_4$ ($s_5 = 0$ in the autonomous case). The iterated Lyapunov technique described above leads to the results which are given in Table I. Since $S_9 = \Phi$ and $\bar{A}_9 \neq \Phi$, the conclusion is reached that this NFSR is unstable. (It must be pointed out that the Lyapunov method is used here mainly to emphasize the analogy with continuous systems and that the same information could be obtained by working out the complete autonomous state diagram. The method here saves some computation owing to the fact that the entire set of states at each level of the diagram is computed simultaneously and only stable states are found.)

[8] J. LaSalle and S. Lefschetz, "Stability by Liapunov's Direct Method with Applications," Academic Press, Inc., New York, N. Y.; 1961.

[9] J. Massey, *op. cit.*, p. 62.

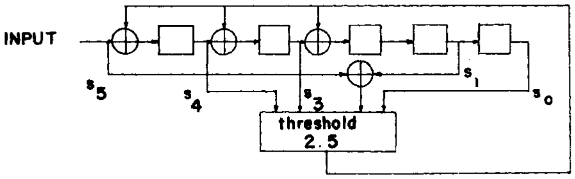

Fig. 2—NFSR for Example.

Driven-Stability

The NFSR of Fig. 2 is unstable, but fortunately it does not follow necessarily that the decoder is subject to indefinitely long error propagation. The reasoning is as follows: The decoder always begins operation with **0** stored in the shift register and is driven into other states by the syndrome input sequence. Thus only those states **s** that can be established by an input sequence should be considered in studying the error-propagation effect. This motivates the following definition:

Definition 2

The NFSR is *driven stable* if and only if, for every state **s** that can be reached from **0** by driving the NFSR with an input sequence, there exists some N such that $\theta^N \mathbf{s} = \mathbf{0}$.

Thus, if the NFSR is driven stable, no decoding error can propagate indefinitely. Conversely, if the NFSR is not driven stable, indefinitely long error propagation is possible. Clearly stability implies driven stability, but the converse is not true. From Table I, it is found that state 23 is the only state which does not go to **0**. (It follows that state 23 must go into itself and is thus also an equilibrium state.) It is readily checked that, under driven conditions, state 23 can be reached only through states 14 or 15 as previous states. State 15 cannot be reached, and state 14 can be reached only through state 28, which in turn cannot be reached. Thus state 23 cannot be established in this NFSR by an input sequence. Consequently, this NFSR is driven stable and a decoding error cannot propagate indefinitely.

TABLE I*
Summary of Calculations for the NFSR of Fig. 2

j	S_j	$\#(\bar{A}_j)$	$L_j(\mathbf{s})$
1	1	30	$(s_4 \oplus 1)(s_3 \oplus 1)(s_2 \oplus 1)(s_1 \oplus 1)s_0$
2	2, 3	28	$(s_4 \oplus 1)(s_3 \oplus 1)(s_2 \oplus 1)s_1$
3	4, 5, 6, 7	24	$(s_4 \oplus 1)(s_3 \oplus 1)s_2$
4	8, 9, 10, 12, 13, 14	18	$(s_4 \oplus 1)s_3(s_1s_0 \oplus 1)$
5	16, 17, 18, 20, 21, 24, 28	11	$s_4[(s_3 \oplus 1)(s_1 \oplus 1) \oplus s_3(s_1 \oplus 1)(s_0 \oplus 1) \oplus (s_3 \oplus 1)(s_2 \oplus 1)s_1(s_0 \oplus 1)]$
6	19, 25, 26, 27, 29	6	$s_4[(s_3 \oplus 1)(s_2 \oplus 1)s_1s_0 \oplus s_3(s_2 \oplus 1)(s_1 \oplus 1)s_0 \oplus s_3s_2(s_1 \oplus 1)s_0 \oplus s_3(s_2 \oplus 1)s_1]$
7	11, 15, 30, 31	2	$s_3s_1[s_4s_2 \oplus (s_4 \oplus 1)s_0]$
8	22	1	$s_4(s_3 \oplus 1)s_2s_1(s_0 \oplus 1)$
9	Φ	1	Conclusion: UNSTABLE

* $\#(\bar{A}_j)$ denotes the number of states in $\bar{A}_j$.

Conclusion

The purpose of this communication is to provide an analytical framework for the study of the error-propagation effect in decoding convolutional codes and to suggest certain methods that appear promising for this study. The important distinction between stability and driven stability of a NFSR was introduced and is a key concept in the study of error propagation. The investigations reported here have raised several interesting questions: Is it true that any "good" decoding algorithm gives a NFSR that is driven stable? What is the relationship between stability and driven stability? What easy tests can be found to determine both types of stability? These and related questions are currently being investigated by the authors.

J. L. Massey
R. W. Liu
University of Notre Dame
Notre Dame, Ind.

A Combinatorial Problem and a Simple Decoding Method for Cyclic Codes

Summary

A relevant problem pertaining to the theory of runs is considered. The solution is given, and in the sequel, a useful identity (Lemma 2) is derived. It is shown how these results apply to decoding of systematic cyclic codes. This leads to a simply implemented error-correcting and detecting decoder. The decoder functions by searching for an error-free string of k consecutive digits. The efficiency of such a decoder is described. The quantitative values are given in Table I. The decoding efficiency is higher when errors occur in bursts, instead of being independently distributed. The use of feedback offers an attractive utilization of the intrinsic error-detecting capability.

Combinatorial Problem

Consider a linear sequence of n binary symbols. Let n_0 of these be zeroes, and n_1 ones. To avoid the trivial cases, assume $n_i \geq 1$, $i = 0, 1$. Then,

$$n = n_0 + n_1 \geq 2. \tag{1}$$

There are $\binom{n}{n_0} = \binom{n}{n_1}$ such distinguishable sequences. For purposes of clarity, let us review the definition:

$$\begin{aligned} \binom{n}{m} &= \frac{n!}{m!(n-m)!} && \text{for } n \geq m \geq 0, \\ &= 0 && \text{for } m > n \geq 0, \quad (2) \\ &= (-1)^m \binom{-n+m-1}{m} && \text{for } m \geq 0 > n. \end{aligned}$$

Next, assume that an integer, $k \geq 1$, is given. Let $N(k, n, n_1)$ denote the number of those distinguishable sequences which do not contain k consecutive zeroes.

Lemma 1

Using the above definitions,

$$N(k, n, n_1) = \sum_{i=0}^{M} (-1)^i \binom{n_1+1}{i} \binom{n-ik}{n_1}, \tag{3}$$

where $M = \min(n_1 + 1, [(n - n_1)/k])$ and the symbol $[R]$ denotes the integer part of the real number R.

Proof: It suffices to give an abbreviated proof here. After all, the problem is suggested as an exercise and the generating function method is outlined in Riordan [1]. In fact, the evaluation of $N(k, n, n_1)$ is tantamount to finding the coefficient of the t^{n-n_1} term in the generating function

$$g(t) = \left(\frac{1-t^k}{1-t}\right)^{n_1+1}. \tag{4}$$

A more recent discussion [2] considers partitioning problems and arrives at similar results.

In attempting to simplify (3) we have arrived at a special result, which occasionally will be useful for quick estimates of $N(k, n, n_1)$. This is, in substance, Lemma 2.

Lemma 2

Let k, n, and n_1 be any non-negative integers. Then

$$\sum_{i=0}^{n_1+1} (-1)^i \binom{n_1+1}{i} \binom{n-ik}{n_1} = 0. \tag{5}$$

Manuscript received October 6, 1963.