working on the top, second,... path of a unique stack. Furthermore, the stack ordering and path extension can be performed in parallel by different machines operating under a central control.

References

[1] J. M. Wozencraft, "Sequential decoding for reliable communication," Sc. D. dissertation, Dep. Elec. Eng., M.I.T., Cambridge, June 1957.

[2] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 260–269, Apr. 1967.

[3] R. G. Gallager, *Information Theory and Reliable Communication*. New York: Wiley, 1968.

[4] K. Zigangirov, "Some sequential decoding procedures," *Probl. Peredach. Inform.*, vol. 2, no. 4, pp. 13–15, 1966.

[5] F. Jelinek, "A fast sequential decoding algorithm using a stack," *IBM J. Res. Develop.*, vol. 13, pp. 675–685, Nov. 1969.

[6] J. L. Massey, M. K. Sain, and J. M. Geist, "Certain infinite Markov chains and sequential decoding," *Discrete Math.*, vol. 3, pp. 163–175, Sept. 1972.

[7] D. Haccoun, "Multiple-path stack algorithms for decoding convolutional codes," Ph.D. dissertation, Dep. Elec. Eng., McGill Univ., Montreal, Canada, June 1974.

[8] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. IT-9, pp. 64–74, Apr. 1963.

[9] J. E. Savage, "The computation problem with sequential decoding," M.I.T. Lincoln Lab., Cambridge, Tech. Rep. 371, Feb. 1965.

[10] I. M. Jacobs and E. R. Berlekamp, "A lower bound to the distribution of computation for sequential decoding," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 167–174, Apr. 1967.

[11] D. D. Falconer, "A hybrid sequential and algebraic decoding scheme," Ph.D. dissertation, Dep. Elec. Eng., M.I.T., Cambridge, Feb. 1967.

[12] F. Jelinek, "An upper bound on moments of sequential decoding effort," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 140–149, Jan. 1969.

[13] G. D. Forney, Jr., "Coding system design for advanced solar missions," NASA, Codex Corp., Watertown, Mass., Contract NAS2-3637, Final Rep., Dec. 1967.

[14] J. K. Omura, "On the Viterbi decoding algorithm," *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-15, pp. 177–179, Jan. 1969.

[15] J. Geist, "Algorithmic aspects of sequential decoding," Ph.D. dissertation, Dep. Elec. Eng., Univ. Notre Dame, Notre Dame, Ind., Aug. 1970.

[16] J. P. Odenwalder, "Optimal decoding of convolutional codes," Ph.D. dissertation, Dep. Elec. Eng., U.C.L.A., Los Angeles, Jan. 1970.

[17] D. Haccoun and M. J. Ferguson, "Adaptive sequential decoding," in *Proc. 1973 IEEE Int. Symp. Information Theory*.

Convolutional Source Encoding

MARTIN E. HELLMAN, MEMBER, IEEE

Abstract—**In certain communications problems, such as remote telemetry, it is important that any operations performed at the transmitter be of a simple nature, while operations performed at the receiver can frequently be orders of magnitude more complex. Channel coding is well matched to such situations while conventional source coding is not. To overcome this difficulty of usual source coding, we propose using a convolutional encoder for joint source and channel encoding. When the channel is noiseless this scheme reduces to a convolutional source code that is simpler to encode than any other optimal noiseless source code known to date. In either case, decoding can be a minor variation on sequential decoding.**

I. Introduction

WE SHOW that convolutional codes can be used at rates greater than one to achieve optimal noiseless source coding and that a type of sequential decoder can be used for decoding such codes. Since the encoder is extremely simple, this technique is well suited to remote telemetry and other applications where encoder complexity must be minimized, even if it is at the expense of greatly increased decoder complexity.

Huffman's optimal technique [1] for noiseless source coding is frequently satisfactory for sources with simple statistics. However, a Huffman code is not easily implemented for a source with long complex memory. Also, such variable length codes require buffers that can dominate the encoding complexity. In contrast, the technique described herein is usable on sources with memory and gives a fixed rate code.

Although the primary interest here is in convolutional source coding for use on noiseless channels, it is easiest to understand the technique when used for joint source and channel coding and when an arbitrary error propagating code is used. The compression problem is a special case within this more general framework and, as we shall see, it is the error propagating nature of convolutional codes that allows their use.

Let us, therefore, for ease of explanation, first consider a system that receives English text from a source, compresses it by 2:1 by removing part of the redundancy, and then uses a rate $\frac{1}{2}$ code, which entails a 2:1 expansion, to correct transmission errors. Overall, the encoder is rate one. It seems somewhat wasteful to remove redundancy only to

Manuscript received June 15, 1973; revised April 21, 1975. This work was supported in part by the U.S. Air Force Office of Scientific Research under Contract F44620-73-C-0065 and in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy, U.S. Air Force) under Contract N00014-67-A-0044.

The author is with the Department of Electrical Engineering, Stanford University, Stanford, Calif. 94305.

add it in for error correction. Of course, the natural redundancy of the source is not used for error correction because it is not well matched to that task.

For example, if the message "I AM NOT ABLE TO PROVIDE SUPPORT." is received as "I AM NOT AGLE TO PROVSDE SUPPORT.", the two errors, indicated by italics are easily corrected. However, a single transmission error can produce an undetected error: "I AM NOW ABLE TO PROVIDE SUPPORT.". If single errors are more likely to occur than multiple errors, the correctable error patterns are not the most probable. In some sense we have chosen the wrong coset leaders.

In this paper we suggest using error propagating codes as a simple means for transforming the natural redundancy of the source into a usable form. Since error propagating codes are usually avoided [2], it may seem somewhat surprising for us to recommend their use. The above example will help to illuminate the motivation behind this suggestion. Suppose the message "I AM NOT ABLE TO PROVIDE SUPPORT." was encoded by a rate one error propagating code. If a single character error occurs, the T of NOT can still be received as a W, but now the remainder of the message will appear garbled: "I AM NOWJ.NXAAVWM, EWTY,ROVBGZ,RI". It is easy to detect that an error has occurred, and without much difficulty it can even be corrected. The most likely possibility is that the J of NOWJ should be a space. Forcing this by a single character correction causes two errors to propagate, and the output will again appear garbled:

"I AM NOW HU.CVKIWXRORBHUWTZHUIGK*".

When all other possible corrections on the J of NOWJ are tried they too yield meaningless output. Assuming a single error, we then try corrections on the W of NOWJ. All except the proper correction cause meaningless output. The code used for this example is described in an earlier paper [3] which discussed using error propagating codes for the detection of errors.

In this paper we show that error propagating codes can be used to correct virtually all errors, so long as the entropy rate of the source is less than the capacity of the channel. We prove this in a general manner that extends to code rates both above one (compression) and less than one. First, we show that convolutional codes are error propagating. Then we use the joint source and channel coding theorem, developed by Gallager for block codes [4, p. 534], [5, p. 162], to prove a similar joint source and channel coding theorem for convolutional codes when used on a discrete memoryless source and channel. This proof is very short, but provides little insight into decoding techniques. We, therefore, outline a proof that shows that a type of sequential decoder can be used. A somewhat heuristic argument is then given which shows that the results extend to more interesting real-life sources.

The technique to be proposed has a tree structure. Jelinek [6] proposed using tree codes for coding with respect to a distortion measure and found very encouraging complexity requirements as compared to block codes.

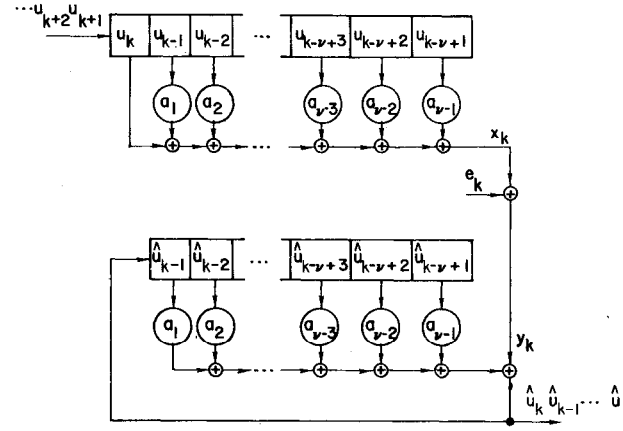

Fig. 1. Rate one binary error propagating code.

However, the technique used here transfers most of the complexity to the decoder, while Jelinek's technique transfers it to the encoder. In some sense, these techniques are duals.

Independently of each other and of this author, two others realized the applicability of convolutional codes to source coding. Priority for the basic idea belongs to them and one contribution of this paper will be to bring additional attention to their work. In an unpublished report, Blizard [7] suggested using a sequential decoding metric that takes the *a priori* probabilities of the source into account. He showed that this allowed data to be decoded at rates above the usual computational cutoff. Koshelev [8], [9] independently came to the same conclusion and derived an H_{comp} for a source. He showed that if H_{comp} is less than R_{comp}, then the expected number of computations per decoded bit was finite. Of course, H_{comp} is greater than H, the true entropy rate of the source.

The approach of this paper is somewhat different since it almost ignores the computational requirements of the decoder, and instead concentrates on showing the optimality of the technique. This position is based on two motivating factors. First, concern here is primarily with the complexity of the encoder. Second, as will be shown later, it appears that the computational burden on the decoder can be greatly reduced by a simple precoding operation. The ideas in this paper are also related to those of Velasco and Souza [10] and of Massey and Ancheta [11].

II. Error Propagating Codes

A binary rate one perfectly error propagating code is an invertible mapping from binary strings $\boldsymbol{u}$ to binary strings $\boldsymbol{x}(\boldsymbol{u})$ of the same length. Both the mapping and its inverse are causal (and hence causally invertible). The crucial property is as follows: Let $\boldsymbol{y} = \boldsymbol{x} \oplus \boldsymbol{e}$, where $\boldsymbol{e}$ is the channel sequence. Then, letting i_0 denote the location of the first error, when the inverse mapping is applied to $\boldsymbol{y}$ the output $\hat{\boldsymbol{u}}$ agrees with $\boldsymbol{u}$ in the first $i_0 - 1$ positions but is totally random after the i_0 position.

In [3], it is shown that the encoder–decoder pair shown in Fig. 1 has the required properties provided $\nu = N$, the length of the binary strings. The encoder is a rate one con-

volutional encoder with the first stage of the shift register tapped with probability one. All other stages are tapped independently, each with probability $\frac{1}{2}$.

Of course, error propagation is perfect only when averaged over the ensemble of codes. However, one can think of choosing a code (i.e., the $\{a_i\}_{i=1}^{N}$) and building an encoder and decoder without telling the receiver which code is being used. Until the first channel error occurs, the receiver is not affected by the choice of code. One time unit after the first channel error, the receiver is affected, but only by the choice of a_1. That is, if the receiver knew the information sequence, it could then compute a_1, but not $\{a_i\}_{i=2}^{N}$, on the basis of the decoded information sequence. At the next time unit, the receiver is affected by a_1 and a_2, etc. Even though we had specified the code, it still is random as far as the receiver is concerned.

The extension to nonbinary error propagating codes is obvious [4, p. 208]. A more interesting generalization is to error propagating codes with rates other than one. Rate $1/n$ codes are constructed in the usual way by transmitting n different mod-2 sums of the shift register contents. The first stage of the shift register is included in each of the n sums, and later stages are included independently and at random in each sum. In the absence of noise the outputs of any one of the mod-2 sums is sufficient for decoding. In the presence of noise the different streams provide independent information about the message.

Rate m codes are constructed by shifting m source bits at a time into the shift register and transmitting one mod-2 sum. The taps are chosen as before. Rate m/n codes are obtained by shifting in m source bits at a time and transmitting n different mod-2 sums.

If the rate of the code is larger than one, then it is not always possible to reproduce the source sequence from the source coded data. However, as shown in Section III, if the rate of the source is less than the capacity of the channel (both measured in bits per second), the decoding will almost surely be performed correctly. Reliable communication is thus possible using these techniques at all rates where it is allowed by any other type of system.

III. Proof of Optimality

We now prove that, for a discrete memoryless source and channel, convolutional codes can be used to perform reliable joint source and channel coding provided that H, the entropy of the source, is less than C, the capacity of the channel, both measured in bits per second. We utilize the following theorem [4, p. 534], [5, p. 162] in the proof.

Theorem 1: Let L source outputs $\boldsymbol{u}$ of a discrete memoryless source with single letter distribution $Q(u)$ be encoded into $\boldsymbol{x}(\boldsymbol{u})$, a sequence of N channel inputs to a discrete memoryless channel with transition probabilities $P(y \mid x)$. Also, within each codeword, let the N letters be chosen independently according to $p^*(x)$, the distribution that achieves capacity. Then if, on an individual basis, each incorrect codeword is independent [4, pp. 206–208] of the correct codeword, the ensemble error probability $P_e(L,N)$ is upperbounded by

$$P_e(L,N) \leq 2^{**}[-\alpha(NC_0 - LH_0)] \tag{1}$$

where C_0 is the capacity of the channel in bits per use, H_0 is the entropy of the source in bits per source output, and $\alpha > 0$. A maximum *a posteriori* probability (MAP) decoder is assumed.

Remark: We are not trying to find the tightest possible bound on error probability. If a tighter bound is desired, then the more general expression for $P_e(L,N)$ given in [4, p. 534] should be used, and optimized over its variable parameters.

We are now in a position to prove the joint source and channel coding theorem for convolutional codes.

Theorem 2: If $R < C_0/H_0$ and MAP decoding is used, then there exist convolutional codes of rate R that can be reliably decoded when used for joint source and channel coding.

Remarks: If the channel is noiseless and of the same alphabet size as the source then $C_0/H_0 \geq 1$, and the convolutional code is operating as a source code at a rate above one. Note that in the terminology used here a rate two code achieves a 2:1 compression. This is consistent with the definition of a rate for error correcting codes but is the inverse of the usual definition of a rate for source codes.

Also, note that convolutional codes lose an additional fraction of their rate through the need for a trailer of known bits to be inserted at the end of the source sequence [12]. As is well known, and as can be seen from the proof below, this additional loss is negligible for long block lengths. This additional loss is, therefore, neglected in defining the rate of the convolutional code. In the proof we take the constraint length of the code to equal the message length plus trailer length plus one. In practice a shorter constraint length will usually be used.

Proof: If $R = m/n$ then m source outputs are shifted into the encoder, and n channel inputs are produced by the encoder, each unit of time. Let us consider an encoder that operates in this fashion for k time units and then shifts in a trailer of mT known symbols for the next T units of time. Letting A denote the size of the source alphabet, overall this produces a tree code of depth $k + T$, with A^m branches emanating from each node at depths between zero and $k - 1$, and one branch emanating from each node at depths between k and $k + T - 1$. Nodes at depth $k + T$ are terminal nodes of the tree. Thus, overall, we encode km source outputs into $(k + T)n$ transmissions, and the actual rate of the code, including trailer loss, is

$$\begin{aligned} R_{\text{act}} &= km/[(k + T)n] \\ &= R/(1 + T/k) \end{aligned} \tag{2}$$

measured in transmissions per source output.

We now upperbound P_e, the block error probability averaged over the usual ensemble of convolutional codes.

In what follows, all probabilities are tacitly ensemble probabilities. First we use the union bound to obtain

$$P_e \leq \sum_{j=0}^{k-1} P_{e,j} \tag{3}$$

where $P_{e,j}$ is the probability that at least one terminal node that first stems from the correct path at depth j has higher *a posteriori* probability than the correct codeword (terminal node).

By starting the convolutional encoder with a one in the first stage we produce a "generalized" coset code, and each codeword has independent identically distributed component letters. By proper choice of the encoder output function we can make this distribution as close to $p^*(x)$ as desired [4, p. 208], and we thus assume $p^*(x)$ to be the distribution on individual codeword letters. It is also well known that, in an infinite constraint length convolutional code, the letters comprising two different codewords are independent except for the letters on the common portion of their paths. The codewords thus meet the conditions of Theorem 1 for bounding $P_{e,j}$. We take $L = (k-j)m$ and $N = (k+T-j)n$. The only problem is that not all source L-tuples are considered. Rather, only the correct source L-tuple (corresponding to the last L source outputs) and those L-tuples disagreeing with the correct L-tuple in the first position are considered in the calculation of $P_{e,j}$. However, $P_{e,j}$ is clearly upperbounded by $P_e(L,N)$ since the latter includes all potential error events of the former. Therefore, combining (1) and (3), we obtain

$$\begin{aligned} P_e &\leq \sum_{j=0}^{k-1} 2^{**}\{-\alpha[(k+T-j)nC_0 - (k-j)mH_0]\} \\ &< 2^{-\alpha TnC_0} \sum_{i=0}^{\infty} 2^{**}[-\alpha i(nC_0 - mH_0)] \\ &= 2^{-\alpha TnC_0}[1/(1 - 2^{-\alpha(nC_0 - mH_0)})] \end{aligned} \tag{4}$$

provided $nC_0 > mH_0$, which is equivalent to the assumption $R < C_0/H_0$.

From (4), we see that P_e can be made as small as desired merely by choosing T to be large enough. R_{act} can then be made to approach R by letting $k \to \infty$ with T fixed.

Q.E.D.

IV. Sequential Decoding

The MAP decoding assumed in the above proof suffers from the usual exponential growth in complexity with the block length. Fortunately, the tree structure of convolutional codes (and of error propagating codes in general) permits the use of a modified sequential decoding algorithm. Because of limitations of space, we only outline the proof. The metric increment along a branch going from a node at depth $j-1$ to a node at depth j is

$$\begin{aligned} m(\boldsymbol{u},\boldsymbol{x},\boldsymbol{y}) &= \ln [Q(\boldsymbol{u})P(\boldsymbol{y} \mid \boldsymbol{x})P(\boldsymbol{y})], \qquad j \leq k \\ m(\boldsymbol{x},\boldsymbol{y}) &= \ln [P(\boldsymbol{y} \mid \boldsymbol{x})/P(\boldsymbol{y})] - mH_0, \\ & \qquad k+1 \leq j \leq k+T \end{aligned} \tag{5}$$

where $\boldsymbol{u}$ is the source m-tuple corresponding to the branch, $\boldsymbol{x}$ is the n-tuple that would have been transmitted if the branch were traversed, and $\boldsymbol{y}$ is the received n-tuple during the jth time unit. As usual, $P(y)$ is defined by

$$P(y) = \sum_x p^*(x)P(y \mid x). \tag{6}$$

We have proved that the usual sequential decoding algorithms, when used with these metrics, allow reliable decoding at all rates $R < C_0/H_0$. The proof is complicated by the nonhomogeneity of the metric increments (since they depend on $\boldsymbol{u}$), but this can be handled by upperbounding P_e by $P(\beta) + \sum_{j=0}^{k-1} EN_j$, where $P(\beta)$ is the probability that the metric drops by more than β units along any portion of the correct path, and EN_j is the expected number of terminal nodes first stemming from the correct path at depth j, and with final metric less than β below the metric of the correct path at depth j. β is a parameter of the algorithm and grows roughly logarithmically in k.

The expected metric increment on the correct path can be shown to equal $nC_0 - mH_0$ and is thus positive for $R < C_0/H_0$. $P(\beta)$, therefore, goes to zero exponentially in β and grows at worst linearly in k. Hence a logarithmic growth of β in k suffices to keep $P(\beta)$ at an acceptably low level.

Upperbounding EN_j is somewhat more difficult because of the nonhomogeneity in the metric. Fortunately, a rather simple bound $EN_j < EN_j'$ works, where N_j' is the number of terminal nodes with metrics above $-\beta$ in a tree with maximal depth $k+T-j$ and with T nonbranching final stages and where all the $\boldsymbol{x}$ are chosen independently of $\boldsymbol{y}$. This is similar to upperbounding $P_{e,j}$ by $P_e(L,N)$ in the proof of Theorem 2 and corresponds to including the correct path, and all paths stemming from it at depth $j+1$ or greater, in the set of potential error causing paths. To compensate for this addition, we make the codewords thus added different from, and independent of, their actual values.

Then, with minor variations on the usual bounding techniques for sequential decoding, it is possible to show that $P_e \to 0$ as $k \to \infty$ provided T grows logarithmically in k.

Example: Consider encoding a memoryless binary source with $p = P(1) = 0.1$ for transmission over a noiseless binary channel. Since $H_0 = H(0.1) = 0.469$ bits/source output and $C_0 = 1$ bit/transmission, the maximum theoretical compression coefficient is 2.132 to 1. If a rate two convolutional code is used there are four paths leaving each node, but there is only one transmitted bit per branch. If the jth received bit disagrees with a branch connecting a node at depth $j-1$ to a node at depth j, that path is assigned a metric of $-\infty$ since then $P(y \mid x) = 0$. If the received bit agrees with the hypothesized transmission, the metric increment m is $\ln [Q(\boldsymbol{u}) \cdot 1/(\frac{1}{2})] = \ln [2Q(\boldsymbol{u})]$. For $\boldsymbol{u} = 00$, $m = \ln [2(0.9)^2] = 0.482$; for $\boldsymbol{u} = 01$ or 10, $m = -1.715$; and for $\boldsymbol{u} = 11$, $m = -3.912$. Normalizing, we find that every occurrence of $u = 0$ adds $+1$ to the metric and every occurrence of $u = 1$ adds -8.109. The expected

metric on the correct path is $(0.9)(+1) + (0.1)(-8.109) = 0.089$, which is positive. The moment generating function of the metric increment per branch on a "typical" incorrect path with half zeros and half ones is

$$g(s) = [(e^s + e^{-8.109s})/2]^2. \quad (7)$$

The minimum of $g(s)$ is near $s = 0.23$ and is $g_{\min} = 0.49950$. That this minimum is below $\frac{1}{2}$ is crucial to the success of the decoder, which is operating at rate two, and thus typically has two branches per node that are not ruled out by the received data. The product of $g_{\min}$ and the number of branches per node must be less than one to guarantee "extinction" of incorrect subtrees.

In practice one would tend to use metric values of $+1$ and -8, or $+10$ and -81 and use integer arithmetic for metric calculation. We implemented such a decoder and found promising computational requirements even at this high rate. However, the experiment was small in scope and cannot be construed as strong evidence of reasonable computational requirements. Indeed, we would expect that in long blocks of data we would see large fluctuations in the required decoding effort since we are operating above R_{comp} [8].

V. Sources with Memory and Predistortion

The proofs given thus far depend on the source being memoryless. In this section, we argue that the simplicity of the encoder need not be greatly increased when dealing with more interesting sources with memory (e.g., video or speech). However, it appears that the complexity of the decoder then becomes astronomical. We explore possible techniques for returning the decoder complexity to the realm of reason without sacrificing too much simplicity at the encoder. However, this is largely conjecture. What is needed is a better understanding of the tradeoffs between encoder and decoder complexity.

As an example of a source with memory, let us first consider the English language since its structure is easily understood. If the decoder uses a metric that corresponds to a first-order approximation to English (i.e., letters are independent but not uniformly distributed), then it is possible to obtain as much compression as if the source really were emitting first-order English. This is because only the distribution of the metric on the correct path is different from that for a true first-order English source with no memory. The distribution of the metric increments on incorrect paths is not affected. Yet, while the metric increments on the correct path will exhibit some memory, if we look at the total metric increment along a large number of successive branches, we will find little difference in behavior from having a true first-order English source. This is because of the quasi-ergodic nature of actual English.

If the decoder uses an n-gram approximation in calculating metric values, then higher compression factors are possible. For large n this compression factor approaches the "true" maximum compression coefficient of English. A similar argument indicates that, in video, the use of a large number of previously decoded neighboring picture elements to generate the distribution on the next element to be decoded should allow as much compression as any other information-preserving compression technique.

It is important that the constraint length of the encoder be long compared to the memory of the source so that high detail areas of a picture or atypical words in English (e.g., QUIZZICAL) can "borrow" redundancy from more redundant portions of the source output. This increases the cost of the encoder, but not significantly since the cost grows only linearly in the constraint length. Of more concern is the exponential growth of computation and, therefore, cost at the decoder as it searches almost exhaustively over these atypical regions.

If the encoder could determine the regions of atypicality in the source output, it could take one of several actions. It could vary its rate and although this would require a buffer, we believe that this technique would have a lower probability of buffer overflow than other techniques for a fixed buffer size and compression coefficient. This is because the decoder can still "borrow" redundancy from more redundant portions of the source output, thereby requiring less variation in code rate.

Another action the encoder could take would be to distort the source output in the atypical regions, making it look locally more typical. For example, in decoding a memoryless binary source with $p = 0.1$, we once used a slightly mismatched metric and found that four bit errors were made in a region with higher than usual density of ones, $\boldsymbol{u} = \cdots 11000000010000000110 \cdots$ to be precise. We could probably have avoided this decoding error by changing any of the five ones to a zero prior to compression. This would have incurred only one bit error instead of four and also would have reduced the decoding effort.

This two-stage source encoding (predistortion followed by compression) has advantages over certain more usual methods. For example, in transform coding [13] of video, the distortion is introduced in an undetermined manner, and detail information is hard to distinguish from noise. With the suggested two-stage technique, the distortion is introduced in a deterministic manner, and this manner can be chosen to preserve informative data. Gray [14] has been led by a different line of reasoning to the same conclusion concerning the desirability of this predistortion process being followed by noiseless coding.

These ideas are also closely related to those of Koshelev [9], who suggested "smoothing" of the source output to ease the computational burden on the decoder. His smoothing operation is a predistortion so that the maximal metric drop on the correct path does not exceed some prespecified level. The distortion is removed by adding a supplementary block of information that corrects the distortion.

Of course, the need to calculate the metric on the correct path (or equivalently to isolate the atypical regions in the source output) increases encoder complexity. For a memoryless source, this is not usually a significant cost (e.g., add

+1 when $u = 0$ and -8 when $u = 1$). However, a source with complex memory will have a complex metric. If the encoder uses a poorer, but simpler, approximation to the source in calculating its metric, it will partially ease the decoding burden. Here is an obvious tradeoff in encoder and decoder complexity. Whether it will lead to practical techniques remains to be seen.

VI. Discussion

It is interesting to note that these codes, used either as source codes or as combined source and channel codes, do not require knowledge of the source statistics at the encoder. For example, when encoding a binary representation of written language, the encoder is the same no matter what the language. It is only at the decoder that the source statistics are needed for calculation of the metric. This indicates that an adaptive strategy might be desirable. Early messages are repeated (or sent with a lower rate code) so that, even with no knowledge of the redundancy, they are decodable. The source statistics are learned from these messages and used in the calculation of the metrics for later messages. Again, such a strategy is well suited to remote telemetry since it is only the decoder that adapts its structure and must, therefore, be programmable.

While we have concentrated on convolutional codes, it is possible to show that any error propagating rate one transformation can be used to effect data compression by deleting an agreed upon sequence of its outputs. For example, deleting every other output yields $R = 2$. Certain cryptographic transformations are error propagating, and this is often listed as a disadvantage [15]. If, however, the deciphering mechanism (human or machine) can distinguish between meaningless and meaningful source outputs it is capable of correcting these transmission errors by the trial-and-error technique described in Section I. Therefore, error propagation should perhaps be listed as a desired property of cryptographic systems; and, to the extent that code clerks have used this technique (out of necessity to decipher the rest of the message rather than out of a compelling desire for accuracy), the technique of using error propagating codes for error correction is hardly new.

"Give me a fruitful error anytime, full of seeds, bursting with its own corrections."[1]

VII. Acknowledgment

The author wishes to thank one of the reviewers for suggesting several improvements, most notably the proof based on the block coding theorem.

References

[1] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proc. IRE*, vol. 40, pp. 1098–1101, Sept. 1962.
[2] J. L. Massey and R. W. Liu, "Application of Lyapunov's direct method to the error-propagation effect in convolutional codes," *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-10, pp. 248–250, July 1964.
[3] M. E. Hellman, "On using natural redundancy for error detection," *IEEE Trans. Commun. Technol.*, vol. COM-22, pp. 1690–1693, Oct. 1974.
[4] R. G. Gallager, *Information Theory and Reliable Communication*. New York: Wiley, 1968.
[5] F. Jelinek, *Probabilistic Information Theory*. New York: McGraw-Hill, 1968.
[6] ——, "Tree encoding of memoryless time-discrete sources with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 584–590, Sept. 1969.
[7] R. B. Blizard, "Convolutional coding for data compression," Martin Marietta Corp., Denver Div., Rep. R-69-17, 1969.
[8] V. N. Koshelev, "Direct sequential encoding and decoding for discrete sources," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 340–343, May 1973.
[9] ——, "Convolutional data compression with preventive smoothing," (in Russian), to appear *Probl. Inform. Transmission*.
[10] F. Dias Velasco and C. deRenna e Souza, "Sequential syntactical decoding," manuscript.
[11] T. C. Ancheta, Jr., "Syndrome-source coding for data compression," in *1974 IEEE Int. Symp. Information Theory*, 1974, p. 64.
[12] R. Johannesson, "On the error probability of general tree and trellis codes with applications to sequential decoding," submitted to *IEEE Trans. Commun. Technol.*
[13] P. A. Wintz, "Transform picture coding," *Proc. IEEE*, vol. 60, pp. 809–820, July 1972.
[14] R. Gray, "Sliding block noiseless source coding," submitted to *IEEE Trans. Inform. Theory*.
[15] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, pp. 656–715, 1949.

[1] Vilfredo Pareto (1848–1923), Italian economist and sociologist for whom the Pareto distribution, so important to sequential decoding, is named. This quote is from a comment on Kepler.