Communications Research Centre

EXPERIMENTAL RESULTS ON THE APPLICATION OF CONVOLUTIONAL CODING AND VITERBI DECODING AS A FREQUENCY DIVERSITY TECHNIQUE IN HF DATA TRANSMISSION

by

B.D. McLarnon

Government of Canada
Department of Communications

Gouvernement du Canada
Ministère des Communications

CRC REPORT NO. 1377

OTTAWA, FEBRUARY 1985

Canada

COMMUNICATIONS RESEARCH CENTRE

DEPARTMENT OF COMMUNICATIONS
CANADA

EXPERIMENTAL RESULTS ON THE APPLICATION OF CONVOLUTIONAL CODING AND VITERBI DECODING AS A FREQUENCY DIVERSITY TECHNIQUE IN HF DATA TRANSMISSION

by

B.D. McLarnon

(Information Technology
and Systems Branch)

CRC REPORT NO. 1377

February 1985
OTTAWA

TABLE OF CONTENTS

EXPERIMENTAL RESULTS ON THE APPLICATION OF CONVOLUTIONAL CODING AND VITERBI DECODING AS A FREQUENCY DIVERSITY TECHNIQUE IN HF DATA TRANSMISSION

by

B.D. McLarnon

ABSTRACT

A method of using coding and soft-decision decoding techniques to replace a conventional diversity combiner is examined. It uses a rate-1/2 convolutional code and Viterbi decoding in a dual diversity configuration at 75 bps, and is easily implemented with an eight-bit microprocessor. The results of comparisons between the coding technique and standard diversity combining on several short- and medium-range HF links are given, and its advantages and limitations in HF data systems applications are discussed.

1. INTRODUCTION

In-band frequency diversity has long been recognized as an effective technique for improving the bit error rate performance of HF data systems. Application of this technique typically takes the form of two low-speed FSK data channels, separated by about one kilohertz and carrying the same information. The output data stream from the demodulator is formed by means of one of the classical combining methods (linear, selection, maximal-ratio). The improvement in bit error rate (compared to a single data channel) experienced in a given situation depends upon the particular method used and on the correlation bandwidth of the channel. In general, the improvement obtained is greatest when the correlation bandwidth becomes less than the channel separation, and maximal-ratio combining usually outperforms the other techniques, although the difference is often insignificant in practice. For two-branch diversity systems, an average improvement in bit error rate of one order of magnitude is often cited [1], and the author's experience would appear to confirm this rough estimate.

Although there is limited scope for improvement of HF

frequency diversity systems, given the limited bandwidth available to most users and the fact that maximal-ratio combining has been shown to be theoretically optimum; however, there remain some areas in which improved performance can be pursued. One such area relates to performance of diversity combiners on channels with interference, which is a highly frequent occurrence on HF channels. Some very promising work has been reported in this area [2],[3]. Another possibility, which is addressed in this report, is to optimize the performance of the combiner with respect to a different criterion than bit error rate.

Viewed from a slightly different perspective, an N-branch frequency diversity system can be considered to be a rate 1/N error-correction code applied in the frequency domain, the code of course being a simple repetition code. Soft-decision decoding is a desirable, and in the case of N=2, essential, ingredient in the error correction process; the various combining techniques are thus equivalent to different soft-decision decoding algorithms. This point of view then leads to the question of whether the use of more powerful error-correcting codes, applied in the frequency domain and decoded using soft-decision techniques, can lead to better performance than conventional frequency diversity.

As the number of simultaneously transmitted tones increases, new possibilities for coding emerge. Transmission of the data in diversity pairs can be replaced with block encoding such that a full codeword is transmitted during each symbol period and decoded with a soft-decision algorithm. Probably the best-known example of this type is the (25,16) block code used with a 25-tone modem in the Codem system [4]. For a simple two-tone system, on the other hand, a rate 1/2 convolutional code is a more obvious candidate. Use of this type of code would result in spreading of the transmitted information in time as well as frequency, which can be advantageous in the highly dispersive HF channel. Very little has appeared in the literature on the application of convolutional codes in this manner; however, a paper by Kohlenberg and Berner [5] giving results comparing dual diversity to a rate 1/2 convolutional code shows much lower error rates for the latter. The code used is not described in detail, but apparently the decoder required was rather complex and the constraint length considerable, since the decoding delay was of the order of 450 bit times.

In the work reported here, it was decided to examine the performance of a convolutional code of more modest constraint length which could be decoded using a simple single-board microcomputer. Since it was to be compared to a maximal-ratio type diversity combiner, it was considered preferable that the decoder not be of substantially greater hardware complexity than such a combiner. The Viterbi decoding algorithm appeared to lend itself quite readily to microprocessor implementation, and some information on

implementation techniques was available in the literature [6,7]; therefore this algorithm was selected for the comparison. Since it is not feasible to use codes with long constraint lengths (i.e.,>10) with this decoding technique, it was not expected that dramatic improvements in bit error rate would result, since the errors on HF channels will often occur in bursts which exceed this length. As mentioned above, however, bit error rate is not the only performance measure which one might seek to optimize. In particular, block error rate performance was of considerable interest to us since we were also engaged in work on new ARQ systems. It was thought that the convolutional code, despite its short constraint length, might provide better performance than the conventional combiner during the intervals between bursts when the errors tend to be more randomly distributed. This in turn may lead to fewer block errors and hence to an increase in throughput of an ARQ system. On the other hand, the Viterbi decoder can be expected to prolong the effects of error bursts while it finds its way back to the correct path. The resulting propagation of errors might well cancel any improvement gained during the periods between bursts. This question could only be resolved satisfactorily by conducting a series of on-the-air tests.

2. THE VITERBI ALGORITHM

This section contains a brief and nonmathematical explanation of the operation of the Viterbi decoding algorithm. Many more comprehensive treatments of the subject are available in the literature (see, for example, [8] or [9]).

The first step in understanding the algorithm is to examine the operation of the convolutional encoder. The example used almost invariably in illustrating convolutional coding is shown in Figure 1. This is a rate-1/2 encoder consisting of a three-bit shift register (with parallel outputs) and two exclusive-OR gates. In spite of its extreme simplicity, this code is actually capable of yielding a useful coding gain, and it was in fact used during some of the HF tests. In general, a k-bit shift register with N exclusive-ORed outputs produces a constraint-length k, rate-1/N binary encoder, the actual code of course depending upon the choice of shift register taps. The quantity k is known as the constraint length since it gives the number of bits in each output data stream which are affected by a particular input data bit as it propagates through the shift register; in other words, k is the degree of time spreading of the transmitted information.

For a given set of taps, the pair of bits outputted by the encoder depends upon the data bit shifted in plus the previous contents of the shift register. The value of the k-1 rightmost bits of the shift register is known as the state of the decoder; in the case of the encoder in the figure, the state

is thus $b_1 b_0$. The most significant bit is not needed in the state description since it has no effect on the encoder output when the next input bit is shifted in. Knowledge of the successive encoder states (plus the tap positions) is sufficient to determine the encoder outputs, denoted T1 and T2 in the figure. At the decoder, one faces the inverse problem: by observing the data streams T1 and T2, now possibly corrupted by bit errors, determine the encoder state transitions and hence the input data stream. This is the problem addressed by the Viterbi algorithm.

Fundamental to the operation of the Viterbi algorithm is the description of the encoder state in terms of a trellis diagram, a portion of which is shown on the righthand side of Figure 1 for the case of k=3. The trellis diagram is similar to a standard state transition diagram but with the added dimension of time. The nodes corresponding to the 2^{k-1} possible states are redrawn, proceeding from left to right, for each successive state transition; the history of the encoder and its input are thus shown by a path traced through the trellis. The path from one state to the next is called a branch. For each state there are two possible transitions to the next state, the upper branch occurring when a zero is shifted into the register, and the lower branch occurring when a one is shifted in. Likewise, there are two possible branches entering each state. The trellis shown in the figure is valid for any binary code with k=3; the differences between codes lies in the encoder output bits associated with each possible branch. The pair of bits generated for a particular branch is of course known to the decoder. The decoder can thus test the likelihood of a branch by measuring how closely the actual received bit pair resembles that which would be expected in the event of that branch occurring. The result of this measurement process is a branch metric for each branch, arrived at by calculating in some fashion the distance between the hypothesized and actual bit pair. This is the point at which "soft decision" of the incoming data bits plays an essential part.

It is a straightforward extension of the branch metric calculation to calculate a path metric for a hypothesized path through the trellis by summing the appropriate branch metrics and scaling the total (in the following we shall adopt the convention that a lower metric corresponds to greater likelihood, since that is the convention followed in the implementation to be described). In this way, the possible paths can be compared in order to determine which is the most likely. It would not be feasible, however, to evaluate all possible paths. A considerable path history is required after a data bit enters the decoder before the encoder can make a reliable decision on the value of that bit. The number of possible paths grows exponentially with trellis depth (i.e., the number of bit times of decoder delay) and would quickly become overwhelming. Fortunately, it is not necessary to retain all paths; one can see this by making the observation that of the two paths

entering each state, the path having the higher metric can be discarded since its metric can never fall below that of the other path now that the paths have merged. This means that only 2^{k-1} paths, or one for each state, need be evaluated, regardless of the path history (decoding depth) used for decoding. These paths are often referred to as the survivor paths.

Once sufficient path history has been accumulated after the start of data transmission, the decoder can begin decoding the data, and thereafter will output decoded bits at the same rate as they are inputted to the encoder. For each new state transition, the decoder must update the metrics for all survivor paths, select the path with the lowest metric, and trace that path back through the trellis to the point at which the state transition caused by the data bit to be decoded occurred. After this bit is determined and sent to the output, the corresponding path history information for all of the survivor paths can be discarded in preparation for decoding the next data bit. A decoding depth of four or five times the constraint length has been shown to provide satisfactory operation [9]. For the k=3 code, then, one must store sufficient information about the paths to allow tracing them back twelve to fifteen bit times.

Although the best way to understand how error correction takes place in the Viterbi decoder is to work through some examples on a trellis diagram using a k=3 code and various error sequences, we shall settle for the following heuristic explanation. Suppose the decoder has been initialized so that all path metrics are zero. Assuming the data is free of errors, the metric for the correct path will remain near zero since all of its branch metrics will be small; the metrics for the other survivor paths, on the other hand, will quickly build up and then fluctuate at substantially higher levels since many of their branch metrics will be relatively large (assuming that a good code has been chosen). Now if bit errors occur, the gap between the metric for the correct path and those of the others will narrow with each error. As long as the correct path remains among the survivors, however, it is probable that no output error will occur, since if one traces back an incorrect path with a momentarily lower metric, it will very likely merge with the correct path before the point at which the bit to be decoded is reached. The real danger lies in the possibility that the correct path might be discarded completely, in which case output errors are inevitable.

To see what type of error event could cause the correct path to be discarded, refer to the trellis diagrams shown in Figure 2. The top diagram shows a larger fragment of the trellis for k=3 than in the previous figure, covering a period of five bit times. Any path progressing from left to right through the trellis constitutes a valid code sequence. Notice that the structure of the trellis is the same for any code with k=3, the only difference between codes being the

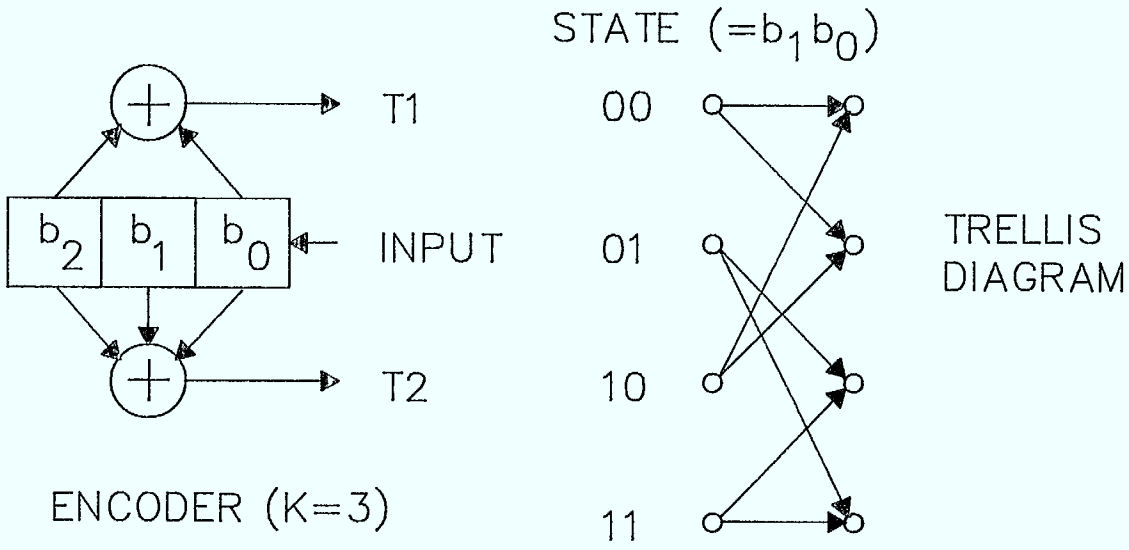

Figure 1. Rate-1/2 Convolutional Encoder and Trellis Diagram

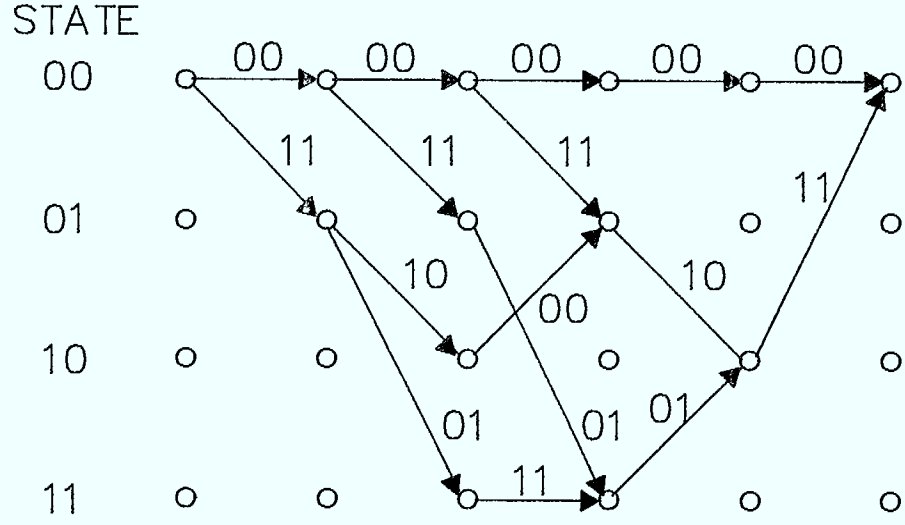

Figure 2. Trellis Diagram Illustrating Error Correction

actual bit sequence generated by the encoder for each branch. Now suppose (without loss of generality) that the correct path through this segment of trellis is the straight line across the top, corresponding to an all zeros data sequence. At the point in time shown on the far right-hand side of the diagram, under what circumstances would the correct path be discarded in favour of another path? For this to occur, another path terminating at the state 00 node would have to have a lower metric. The minimal error patterns which would cause this to happen correspond to those paths which differ the least from the correct path. These paths are shown in the lower diagram of the figure, where all other possible paths have been removed for clarity. Also shown in this diagram are the encoder output bit pair (T1,T2) corresponding to each branch for the particular encoder configuration of Figure 1. The alternate path which differs least from the correct path is the one which diverges from it three nodes prior to the end, corresponding to the code sequence 0000111011 during the time interval shown. This has a Hamming distance of five from the correct sequence; this is in fact the so-called minimum free distance for this code, which happens to be optimum for k=3. Three other alternate paths are shown, two with distance six and one with distance seven. Thus it can be seen that, in terms of hard decisions, the decoder can correct up to two errors (within the constraint length of three information bits) since it would take at least three bit errors to push the metric of the correct path above that of an incorrect one. Soft-decision decoding improves the performance further, making it possible to correct up to four errors under some circumstances.

3. DECODER DESCRIPTION

The operation of the Viterbi decoder can be described in terms of five functional blocks, plus supervisory timing and control functions. They are depicted in Figure 3, and are described briefly below:

(1) Data Acquisition - This function is driven by the recovered clock signal, which provides a rising edge coincident with the center (maximum eye opening) of the data bit. The eye openings for each of the two data bits corresponding to the two branches of the encoder output are sampled and digitized in rapid succession. The usual requirement for branch synchronization (i.e., distinguishing between the data streams T1 and T2) is avoided by the use of separate FSK subchannels rather than interleaving for the transmission of the two data streams. In order to accommodate experimentation with different quantization levels and nonlinear detection characteristics, the actual sample values used for computations were determined by means of a lookup table. Provision was also made for realigning the two bit streams if there were time offsets between them.

(2) Branch Metric Computation - From the pair of soft-decision sample values derived as described above, a metric is computed for each of the four possible pairs of data values which might have been transmitted. In all of the tests to be described here, sixteen-level quantization was used for the sample values, so that a value of 0 corresponded to a reliable received zero bit and 15 to a reliable received one bit. The metric for a hypothetical received bit is then the difference (absolute value) between the ideal value for that bit and the actual received sample value, and the branch metric for a pair of bits is simply the sum of the individual bit metrics. For example, if the received bits had the quantized values 4 and 13, and the hypothesized transmitted pair were zero and one, respectively, then the metric for this particular branch would be (4-0)+(15-13)=6. Branch metrics therefore range in value from 0 to 30.

(3) Path Metric Updating and Storage - As noted previously, for each state of the decoder (i.e., each node in the trellis diagram), there are two possible paths entering that state. Each path has an associated path metric which is obtained by summing the branch metrics for the branches traversed by the path. When a new pair of data bits are received, the corresponding four branch metrics are used to update the path metrics, and the path with the higher metric entering each state is discarded. In order to prevent overflow, the metrics for the surviving paths must be scaled; this is done by subtracting the value of the lowest metric from all of the metrics.

(4) Information Sequence Updating and Storage - In addition to the metric for each survivor path, the encoder must store the data sequence which, when inputted to the encoder, would have resulted in that particular path through the trellis. In principle, this requires nL bits of storage, where $n=2^{k-1}$ is the number of possible encoder states (and therefore the number of survivor paths) and L is the decoding depth. Each time a new pair of code bits is received, it is necessary to store a bit for each possible decoder state; this bit value points to the state which, according to the branch metrics, is the most likely previous state. For example, if one is at state 01, the previous state must be either 00 or 10. If the branch from 00 to 01 has the lower metric, we store a 0; otherwise we store a 1. The array of bits thus stored allows traceback along the path to the required depth. During these tests, the decoding depth was fixed at five times the constraint length, in accordance with the guideline mentioned earlier. For the largest constraint length used, k=7, the storage requirement was then 64*35 = 2240 bits. In order to simplify and speed up the traceback process, however, a full eight-bit byte was used for each bit of path storage.

(5) Decoded Data Output - After the initial period (5k bit times) during which it is only storing data, the decoder must begin outputting decoded data bits. During the path metric

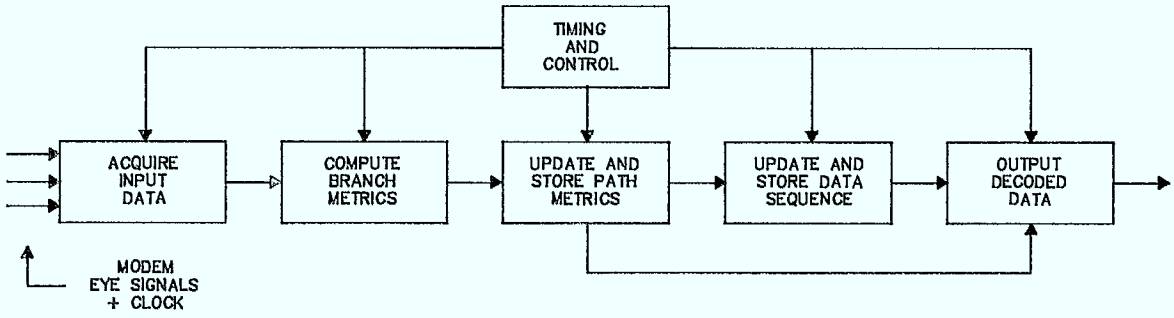

Figure 3. Viterbi Decoder Block Diagram

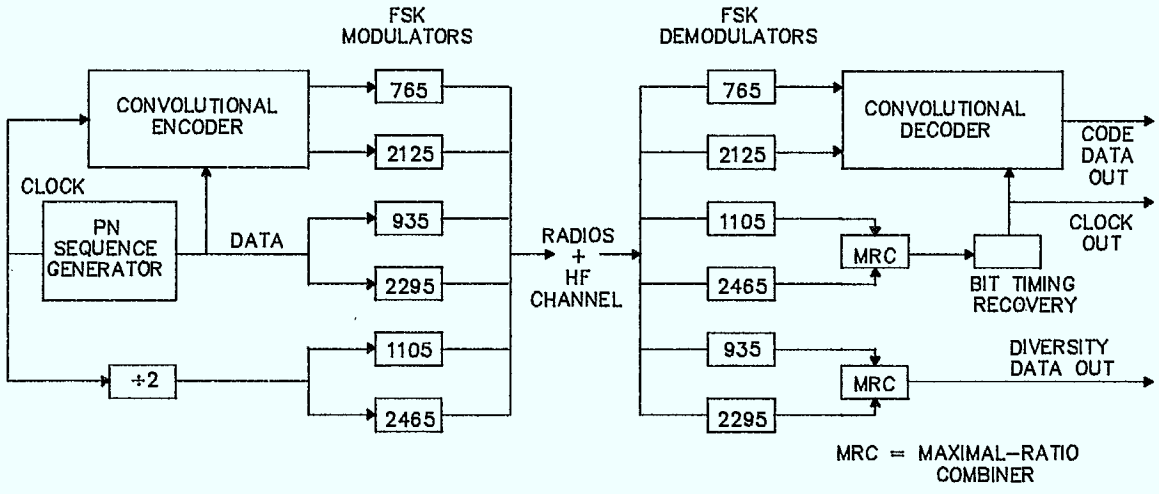

Figure 4. Experimental Set-up for On-the-air Tests

updating procedure described above, the path with the lowest metric is noted; when the time comes to output a data bit, then, it is only necessary to trace back that path through the path storage array to find the required bit. After the bit is outputted, the corresponding bits for all of the survivor paths can be discarded.

The decoder was implemented on an Intel SDK-85 single-board computer (8085 processor, 3 MHz clock). The a/d converter, multiplexer, and associated signal conditioning circuitry were installed in the prototyping area of the board. The memory requirements for the decoder turned out to be quite modest: about 1.5 Kbytes of ROM for the executable code and data tables, and about 2.5 Kbytes of RAM, mainly used for path storage. Thus implementation of a dedicated Viterbi decoder using a single-chip microcomputer plus perhaps a half-dozen additional chips is certainly feasible for low bit-rate systems. Little attempt was made to optimize the decoder software for speed, since our initial implementation achieved, by a small margin, our objective of decoding a constraint length 7 code at 75 bps. As indicated by Rashvand [10], however, bit rates of 300 bps or more are achievable for such a code, by means of careful software design, higher clock rates, and more powerful microprocessors.

4. EXPERIMENTAL CONFIGURATION

The experimental set-up for comparison of the convolutional code technique with conventional diversity is shown in Figure 4. Six standard FSK channel units (75 bps, 85 Hz shift) were employed, with centre frequencies as shown in the figure. The frequencies were grouped into three pairs, each with 1360 Hz spacing; two pairs carried the data generated by the systems under test, and the third carried an alternating bit pattern which was used to drive a bit timing recovery circuit at the receive end. Maximal-ratio combiners, using the popular "ratio-squarer" circuit [11] were used, except in the case of the pair carrying the convolutional coded data. In the latter case, the AGC portion of the diversity combiner was retained, but two "eye pattern" outputs suitable for soft-decision decoding were provided rather than a single combined output. Not shown is the HF radio equipment: SSB transceivers in the 100 watt class, and broadband dipole antennas. Average transmitter power output was in the 10-30 watt range for most tests. The receiver baseband output signal was recorded on an instrumentation tape recorder for later playback through the receive modems and associated decoding equipment. The data and clock outputs were then routed to the error-counting instrumentation (Hewlett-Packard 1645A), the outputs of which were logged into a desktop computer for storage and future analysis. The transmitted data sequence was a 63-bit pseudorandom pattern.

Three different convolutional codes, with constraint lengths 3, 5, and 7, were tried during the tests. The codes were selected on the basis of optimum distance properties. The shift register taps required to generate a particular code are usually denoted by a pair (for a rate-1/2 code) of numbers N1/N2; the numbers N1 and N2 gives the bits of the shift register, in octal notation, which are exclusive-ORed to produce the output bit streams T1 and T2, respectively. For example, the code produced by the generator of Fig.1 is denoted by 5/7. This is the k=3 code used; the others were 35/23 and 171/133 for k=5 and k=7, respectively.

Two series of on-the-air tests took place during the summer of 1983. The first series involved transmissions over a short-range link between Low, Quebec and Ottawa, a distance of about 60 km, using various frequencies in the 3-8 MHz range. Although a weak groundwave component is observable on this link, it was insignificant compared to the skywave component on the frequencies used. Multipath spreads of 1-3 ms have often been recorded when oblique sounders were operated on this circuit. The second series of tests took place after an opportunity arose to make transmissions from a ship operating off the east coast of Canada, in conjunction with another HF experiment. Its voyage took it from Quebec City into the high Arctic, allowing tests to be carried out over distances ranging from about 400 to 2500 km. During the latter part of this test period, the HF link traversed the auroral belt and rapid fading was often present.

5. EXPERIMENTAL RESULTS

One of the facilities which was included in the program written for analysis of the data logged into the desktop computer was the ability to display the error patterns on the CRT display or printer, so that the patterns from two different tests could be examined side-by-side. When the corresponding data for diversity and convolutional coding were examined in this manner, it was immediately evident that there was a qualitative difference in the error patterns. Both sets of data exhibited the burstiness characteristic of the HF channel; however, in the case of diversity, the transition between bursts and the periods of lower error rates seemed relatively gradual. The errors appeared random much of the time, with frequent isolated single errors. The data from the Viterbi decoder, on the other hand, presented quite a different appearance: dense bursts with relatively abrupt beginning and end, longer error-free gaps, and an absence of single and double errors. The bursts tended to be longer than those in the diversity output, since the decoder required some time to recover and find its way back to the correct path through the trellis. Two typical segments of data, with the bit errors highlighted, are

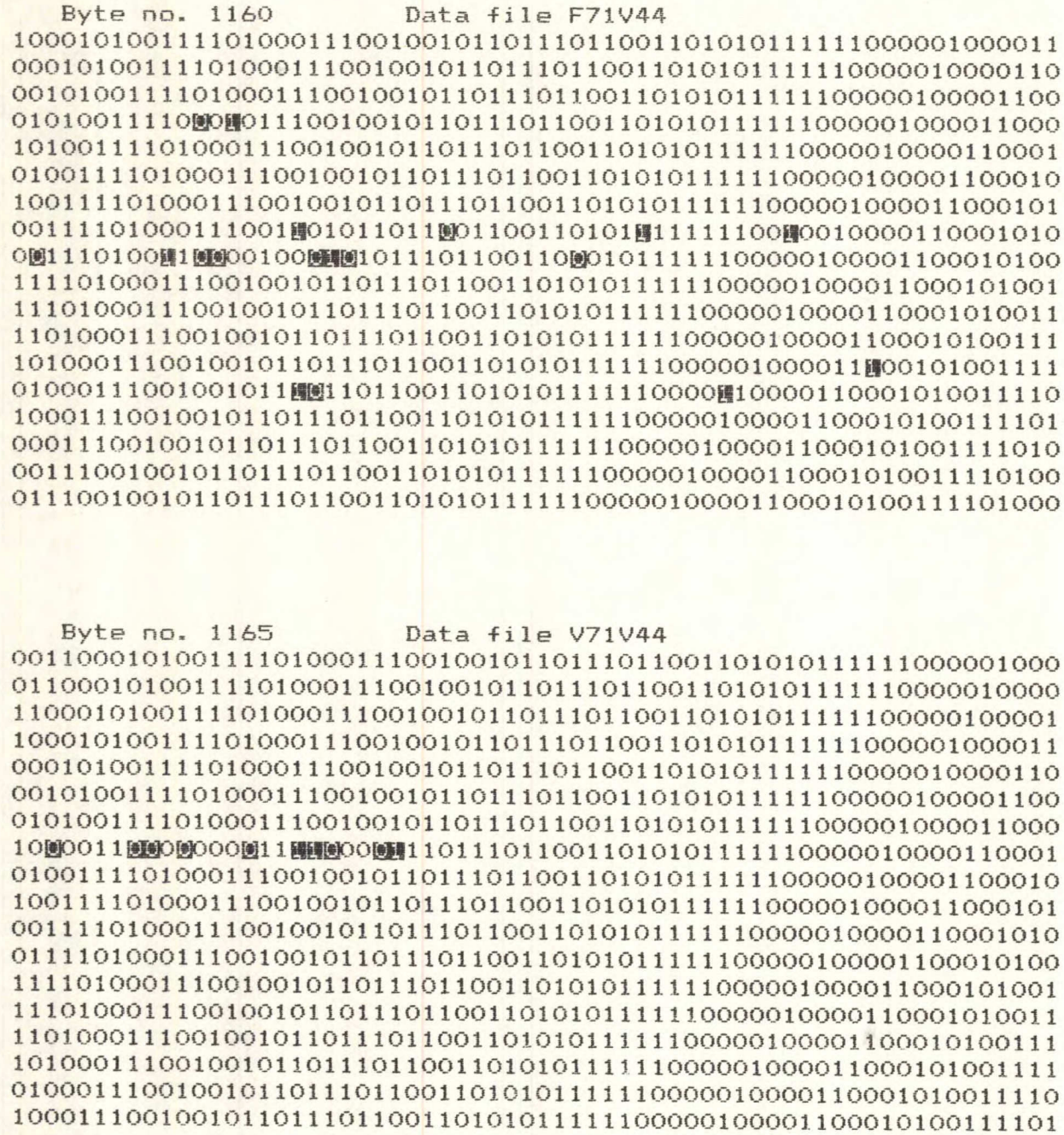

Figure 5. Typical Error Patterns in Diversity (top) and Decoder Outputs

shown in Figure 5. The two segments were collected simultaneously under conditions where the errors were caused by wideband noise. Both systems under test were therefore subject to the same disturbance; the effect on the output error patterns, however, is markedly different in the two cases.

For the purposes of comparison, the tests were divided into segments of approximately five minutes, and the average bit and block error rates were calculated for each segment. As expected, the convolutional code technique did not fare well when compared with dual diversity when the basis of comparison was bit error rate. Although bit errors were clearly being corrected, the number of corrections were often exceeded by those generated by the propagation of errors in the decoder at the end of a burst. In some segments, the decoder output contained more than twice as many errors as the diversity output. Only in those instances when the bit error rates were low (i.e., better than about 10^{-3}) did the channel with coding consistently outperform the diversity channel.

As we had hoped, when the block error rates of the two techniques were compared, the results were much more impressive. The conditions pertaining to the tests are shown in Table 1, and the block error performance for dual diversity and for the various convolutional codes is summarized in Table 2. Two block sizes, 128 and 512 bits, were selected as being representative of those which are appropriate for HF channels. The 128 bit block size was of particular interest to us since it approximates that used in an HF data terminal developed at CRC [12,13].

In addition to the block error tabulation, the table includes the percentage increase in probability of receiving an error-free block for the coding technique versus diversity. This figure represents an upper bound (i.e., disregarding overhead bits, increased turnaround time, etc.) on the improvement in throughput in an ARQ system attainable using the new technique. The improvement did not appear to be a strong function of constraint length, since the results for k=5 were similar to those for k=7, and even the very short k=3 code provided some improvement. However, the small amount of data gathered for the shorter codes does not permit drawing any firm conclusions regarding performance as a function of constraint length. The results of the short-range tests were similar to those of the ship-to-shore medium-range tests, with the block error rates and improvement factor being somewhat lower in the latter case. The improvement factor increased with block size; this is intuitively reasonable since the larger the block, the less likely it becomes that an increase in the duration of an error burst (due to the Viterbi decoder) will mutilate additional blocks.

Not unexpectedly, the difference in block error rates (five-minute average) between the two schemes varied widely, as

shown in the scatter plots of Figures 6 and 7. The dashed lines on the graphs correspond to various throughput improvement factors; for example, a factor of 2.0 means that the coding technique had a probability of delivering an error-free block which was twice that of the standard diversity technique. The improvement obtained ranged from dramatic to insignificant. In some instances the diversity transmission was virtually error-free itself, and thus there was little room for improvement; in other cases, the channel was so poor that neither system could provide a useable error rate. One can observe, however, that in no instance was the performance of the coding scheme significantly worse than that of the standard diversity system.

TABLE 1 - Summary of test conditions

Test Number	Block Size (bits)	Code Length	Total Bits (per mode)	HF Link
1	128	k=7	1,430,000	short
2	128	k=7	506,000	medium
3	128	k=5	352,000	both
4	128	k=3	176,000	both
5	512	k=7	1,430,000	short
6	512	k=7	506,000	medium
7	512	k=5	352,000	both
8	512	k=3	176,000	both

TABLE 2 - Summary of test results

Test Number	Block Error Rate: Diversity	Conv. Code	Throughput Improvement (%)
1	.293	.201	13.0
2	.217	.127	11.5
3	.321	.227	13.8
4	.165	.138	3.2
5	.548	.406	31.4
6	.378	.223	24.9
7	.570	.420	34.9
8	.259	.205	7.3

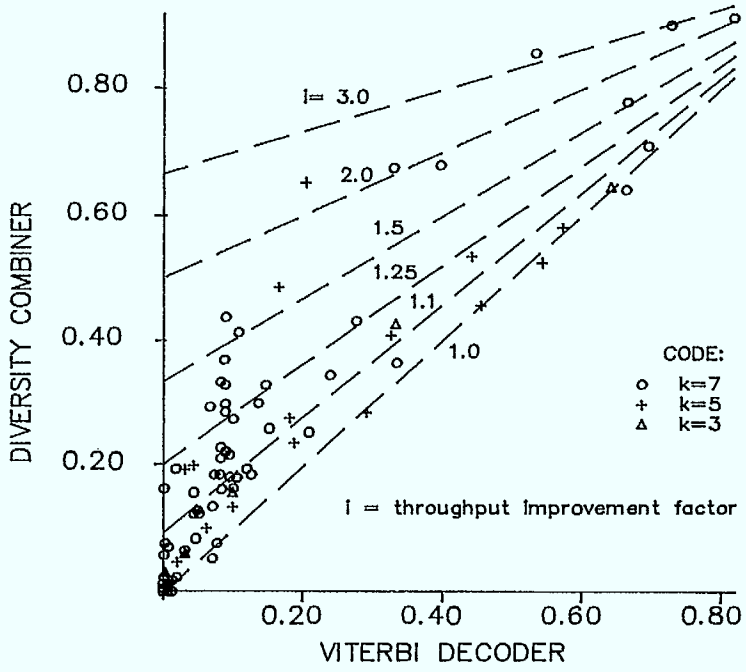

Figure 6. Block Error Rate Comparison, 128 Bit Blocks

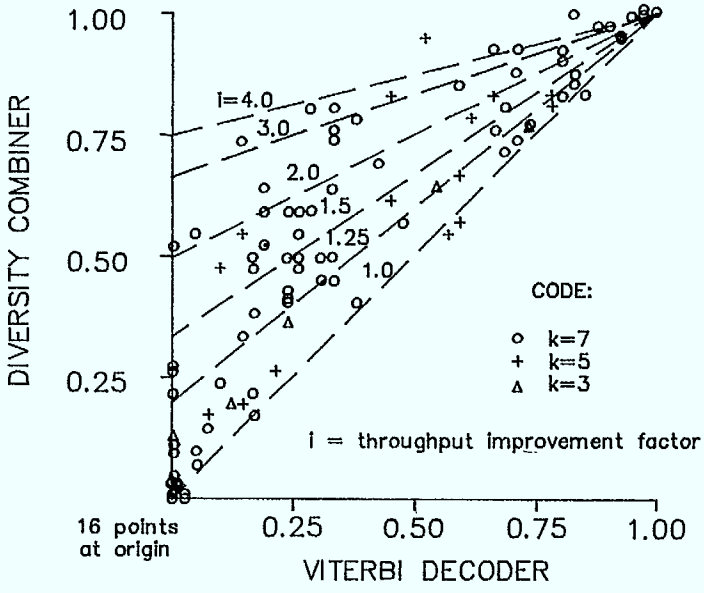

Figure 7. Block Error Rate Comparison, 512 Bit Blocks

6. IMPLEMENTATION CONSIDERATIONS

The coding technique outlined here has a number of practical limitations and clearly would not be appropriate in many situations. It cannot be regarded as a general-purpose diversity-combining method; it is incompatible, for example, with asynchronous data transmission systems. The method is potentially useful for ARQ systems using synchronous transmission provided that the transmissions are not so short that the improvement in throughput is nullified by the increase in overhead bits required for proper operation of the Viterbi decoder. The primary source of overhead is the preamble of approximately 4k bits (where k is the constraint length) needed at the beginning of the transmission in order to initialize the decoder [14]. Also required is a postamble of k-1 bits to bring the encoder to a known state at the end of the transmission. This allows the decoder to immediately select the path leading to that state and thus determine the remainder of the data bits. The data transmission system to which it is attached would most likely be responsible for informing the decoder when this point is reached; alternatively, the postamble could be lengthened to about 4k bits. In the latter case, the decoder design is less system-dependent but throughput efficiency is less than optimum.

For the k=7 code, the minimum overhead is about 34 bits; therefore the transmission length in a half-duplex ARQ system must be several hundred bits if the advantages of the coding technique are to be fully realized. At low bit rates such as 75 bps, however, blocks of this length may not be well matched to the characteristics of the channel and hence the throughput compared to shorter blocks and no convolutional coding. The solution to this dilemma is to adopt a selective-repeat ARQ protocol whereby a number of data blocks are grouped together into one transmission. The block size can then be selected to optimize throughput, taking into account the overhead (such as CRC bits) on each block, and the transmission made sufficiently long that the overhead needed for Viterbi decoding becomes negligible. An example of the use of this type of protocol is the HF data terminal mentioned previously [12,13]. This terminal operates at 75 bps using a dual diversity FSK modem and transmits eight 144-bit data blocks during each transmission, each block having its own error-detection coding. With its total transmission length of 1224 bits, this system would be a good candidate for application of the coding technique (this has not been attempted as yet).

7.0 CONCLUSIONS

It has been demonstrated that the error-correction capabilities of frequency diversity systems can be improved upon in some HF applications by the use of short convolutional codes and soft-decision decoding. The decoding (and encoding)

functions can be performed by common eight-bit microprocessors plus a few support chips. The characteristics and requirements of the overall system must be considered in order to determine the appropriateness of a particular approach. The rate-1/2 convolutional code used with a dual modem configuration and Viterbi decoding provides a hybrid time/frequency diversity capability and is particularly suited to ARQ systems using long blocks or selective-repeat protocols.

8.0 REFERENCES

1. McManamon, P., and Janc, R., 1968, "An experimental comparison of nondiversity and dual frequency diversity for HF FSK modulation", IEEE Trans. Comm. Tech., COM-16, 6, 837-839.

2. Gartell, T., and Cawsey, D., 1978, "An intelligent diversity system for improved signal quality", Proceedings of the Conference on Communications Equipment and Systems, Birmingham, 234-236.

3. Gott, G.F., Dutta, S., and Doany, P., 1983, "Analysis of HF interference with application to digital communications", IEE Proc., 130, F, 5, 452-458.

4. Chase, D., 1973, "A combined coding and modulation approach for communication over dispersive channels", IEEE Trans. Communications, COM-21, 3, 159-174.

5. Kohlenberg, A., and Berner, A.S., 1966, "An experimental comparison of coding vs. frequency diversity for HF telegraphy transmission", IEEE Trans. Comm. Tech., COM-14, 4, 532-533.

6. Rader, C.M., 1981, "Memory management in a Viterbi decoder", IEEE Trans. Communications, COM-29, 9, 1399-1401.

7. Conan, J., 1983, "An F8 microprocessor-based breadboard for the simulation of communication links using rate 1/2 convolutional codes and Viterbi decoding", IEEE Trans. Communications, COM-31, 2, 165-171.

8. Forney, G.D., 1973, "The Viterbi algorithm", Proc. IEEE, 61, 3, 268-278.

9. Clark, G.C., and Cain, J.B., 1981, "Error-Correction Coding for Digital Communications", Plenum Press, New York.

10. Rashvand, H.F., 1982, "Implementation of microprocessors in trellis decoding of convolutional codes", Electronics Letters, 18, 3, 121-123.

11. Kahn, L.R., 1954, "Ratio squarer", Proc. IRE (Correspondence), 42, 1704.

12. Serinken, N.M., and Chow, S.M., 1983, "High frequency data message terminal", Proceedings of the 15th Offshore Technology Conference, 415-418.

13. Bode, L., Chow, S.M., Serinken, N., Tepper, B., and Yamanoto, R., 1984, "Experimental evaluation of a high frequency radio data terminal in the ship/shore mode", CRC Report No. 1375, Department of Communications, Canada.

14. Heller, J.A., and Jacobs, I.M., 1971, "Viterbi decoding for satellite and space communication", IEEE Trans. Comm. Tech., COM-19, 5, 835-848.

ACKNOWLEDGEMENT

The author gratefully acknowledges the contributions of S.M. Chow, for the initial concept and many useful discussions, and D. Dillabough, for the software design of the encoder and decoder.