EVALUATION OF CONVOLUTIONAL CODES BY ERROR TYPES SIMULATION

Shrinivas Gadkari and Kenneth Rose

Department of Electrical and Computer Engineering
University of California, Santa Barbara, CA 93106, USA

ABSTRACT

Error Types Simulation (ETS)- a new method for performance evaluation of convolutional codes, is presented. The key idea is to consider types of channel error sequences separately and estimate the contribution of each type to the information bit error rate (I-BER). By averaging these contributions, weighted by the corresponding probability of occurrence of the type, we obtain the I-BER curve. We show that this approach yields an accurate estimate of the *entire* I-BER curve, while maintaining a computational complexity that is only a very small fraction of the complexity of a Monte-Carlo simulation for a *single* channel condition. ETS is shown to outperform competing methods, in terms of both accuracy of the estimate, and the total computational complexity.

1. INTRODUCTION

Monte-Carlo simulation is often used for performance evaluation of a convolutional code. In Monte-Carlo simulation, we simulate the codec with the given channel conditions and obtain an estimate of the resulting I-BER. The simulation is stopped when sufficient number of errors have been accumulated to estimate the I-BER with the prescribed reliability. For example, an accuracy of $\pm 20\%$ about the mean with 95% confidence, requires accumulation of over 100 decoding errors (assuming a rate $1/n$ code). To get a rough idea of the computational complexity of a typical simulation, consider evaluating the performance of a rate $1/2$ convolutional code producing an I-BER of 10^{-6}. To accumulate 100 decoding errors, we need to simulate the encoding and decoding of approximately 10^8 bits. Moreover, one is often interested in the I-BER of the code under a variety of channel conditions. This is especially true in the case of wireless communication channels which are highly non-stationary and where the channel bit error rate (C-BER) varies over a large range. Thus, if we require to test the performance of several competing codes under various channel conditions, while meeting the stringent requirements on the accuracy of the estimate, the simulation time can be prohibitively long. This motivates the search for an efficient, yet reliable, technique for evaluating the performance of the Viterbi decoder.

This work was supported in part by National Science Foundation under grant no. NCR-9314335, University of California MICRO program, DSP Group, Inc., Echo Speech Corp., Moseley Associates, Qualcomm, Inc., Rockwell International Corp., Speech Technology Labs, and Texas Instruments, Inc.

This problem was addressed in the work of Herro and Nowack [1] where the concept of importance sampling was applied to speed up the simulations. The basic idea was to increase the frequency of occurrence of the "important" events (viz. those that lead to decoding errors), and then appropriately weight the observed simulation data in order to obtain I-BER for the true channel conditions. However, the computational gains obtained by the technique are quite small, and the method is suited only for codes with short constraint length. A different approach to apply importance sampling to Viterbi decoder simulation, called error event simulation (EES), was introduced by Sadowsky [2]. This technique provides a significant reduction in complexity, especially for simulation of codes with low I-BER. However, the estimate of I-BER is obtained for a single C-BER, and not for an entire range of C-BERs. Further, the technique does not produce accurate I-BER estimates for moderately noisy to very noisy channels.

In this paper we take as our starting point the early work of Herro and Nowack, and develop the error types simulation (ETS) method. We classify the channel error sequences according to their *type*, and consider only those types that lead to decoding errors. The idea of importance sampling is used to estimate the contributions of each type to the I-BER. We average these contributions by weighting each term with the probability of occurrence of the respective type at prescribed levels of channel noise. We show that this approach results in an efficient technique for generating the entire I-BER

0-7803-3336-5/96 $5.00 © 1996 IEEE

curve with high accuracy.

2. VITERBI DECODING OF CONVOLUTIONAL CODES

The encoder of a rate k/n convolutional code is a finite state machine which takes in k information bits and produces n channel bits per time unit. The output of the encoder at any time unit depends on the k input bits as well as the past $L-k$ input bits. The state diagram of this finite state machine can be expanded in time to obtain a trellis diagram, for details see eg., [3]. The output bits are transmitted over a noisy channel. The decoder processes the output of the channel and produces an estimate of the input information bits. This estimate is obtained using the Viterbi algorithm to search the trellis for the most likely transmitted sequence. In this paper we will assume that the Viterbi decoder operates using a sliding window, i.e., the decoder makes its decision about the k information bits at time i, by observing the channel output till time $i+M$. A value of M of $\approx 5L/k$ is empirically known to yield a near optimum decoding [4]. Clearly the decoding errors at time i will be affected by channel errors in at most M future time units. As in [1] we further assume that decoding errors at time i are affected by channel errors in at most M past time units.

Consider a sequence of channel outputs, $2M+1$ time units long, consisting of $N = n(2M+1)$ channel bits. We define the number of channel errors in this sequence, as the type of the sequence[1]. Clearly, there are $N+1$ possible types of sequences of length N, including the 'all-0' (error free) type. Let $E(w)$ be the expected bit error rate at time i, when we have type-w channel error sequence, in the time interval $[i-M, i+M]$. Hence the I-BER at time i, over a binary symmetric channel with C-BER ϵ is :

$$p_b = \sum_{w=\lceil d_{free}/2 \rceil}^{N} P_\epsilon(w) E(w), \tag{1}$$

where, $P_\epsilon(w)$ is the probability of w channel bit errors in a sequence of length N bits. Note that p_b is time-invariant and does not depend on i. The quantity $P_\epsilon(w)$ can be evaluated for a binary symmetric channel (BSC), as

$$P_\epsilon(w) = \epsilon^w (1-\epsilon)^{N-w}.$$

From (1) it is clear that to evaluate p_b we only need to obtain an estimate of the quantities $\{E(w)\}$. It is for this purpose that we invoke importance sampling. Observe that once the set of quantities $\{E(w)\}$ has been estimated, the evaluation of (1) for any value of C-BER ϵ, can be done with negligible computational effort. Hence we can obtain the BER curve for an entire range of channel conditions.

[1] Usually the type of a binary sequence is defined as the relative frequency of 1's in the sequence, see eg. [5]. However in the present context it is more convenient define types without normalization

3. ESTIMATION OF $E(W)$

To estimate the quantity $E(w)$, we use the Monte Carlo simulation. Since the code is linear, without loss of generality we can assume that an all zero sequence was transmitted and decode the corresponding received sequence. However, for each value of w, we use a value of ϵ that will produce a channel error sequence of type w with the highest probability. Hence, to estimate $E(w)$, we run the Monte-Carlo simulation with a $\epsilon = \frac{w}{N}$. For each time unit i, we obtain the number of decoding errors and also determine the type of channel error sequence in the time interval $[i-M, i+M]$. The simulation is stopped when we have accumulated sufficient decoded bit errors for channel error sequences of type w. Now, we adjust ϵ to $\frac{w+1}{N}$, and estimate $E(w+1)$. Note that when the simulation was being performed at $\epsilon = \frac{w}{N}$, channel error sequences of type $w+1$ (and other types too) were also generated (along with the corresponding information bit errors). We use this data in addition to the new data that will be generated at $\epsilon = \frac{w+1}{N}$. The number of errors that we require to accumulate depends on the accuracy and confidence interval of the required estimates. If all the estimates of $E(w)$ are within $\pm 20\%$ of the corresponding mean with 95% confidence, we are assured that p_b is accurate with at least this precision.

4. THE CASE OF SMALL W

By simulating the channel with ϵ tailored to the generation of channel error sequences of type w, we significantly reduce the simulation time. Further gains in computations are possible for sequences with weights close to $\lceil d_{free}/2 \rceil$, where d_{free} is the free distance of the convolutional code. We note that for these types of channel error sequences, the decoding errors occur only when the channel errors occur in a small interval around the decoding time i. To see this consider decoding errors caused by a channel error sequence of type $w_0 = \lceil d_{free}/2 \rceil$. Let L_0 be the maximum length of the paths[2] with weight d_{free}. Clearly, if any of the w_0 channel errors lie outside the interval $[i-L_0+1, i+L_0-1]$, we

[2] A path is a sequence of channel output bits produced by the encoder, when it leaves the all-zero state remerges with it for the first time, see for eg. [3]

w	$l_{max}(2w)$	$\tilde{E}(w)$	$E(w)$
4	8	0.0032	10^{-5}
5	15	0.0023	$7.7 * 10^{-5}$

Table 1: The data used to estimate $E(w)$ for types $w = 4, 5$.

ϵ	I-BER (ETS)	I-BER (EES)	I-BER (True)
0.02	$4.0 * 10^{-5}$	$3.9 * 10^{-5}$	$4.2 * 10^{-5}$
0.04	$0.12 * 10^{-2}$	$0.98 * 10^{-3}$	$0.12 * 10^{-2}$
0.05	$0.38 * 10^{-2}$	$0.27 * 10^{-2}$	$0.36 * 10^{-2}$
0.083	$0.41 * 10^{-1}$	$0.20 * 10^{-1}$	$0.41 * 10^{-1}$

Table 2: Here we compare the performance of the ETS method with the EES and the true I-BER.

will not have any decoded bit errors at time i. Hence, for these types the "important" events are the ones where channel errors are localized in a small interval around the time i.

We suggest the following technique for estimating $E(w)$. Let $l_{max}(2w)$ be the maximum length of paths with weight $\leq 2w$. We generate channel error sequences with exactly w errors and $2l_{max}(2w)-1$ time units long. We now decode these individual sequences with the decoder initialized in the all zero state and terminate the decoder at the end of $2l_{max}(2w)-1$ units in the all zero state. The average decoded bit error rate in the time unit $l_{max}(2w)$ can be estimated from these sequences, we denote this estimate as $\tilde{E}(w)$. It can be easily shown that $E(w)$ and $\tilde{E}(w)$ are related as

$$E(w) = \sum_{w'=\lceil \frac{d_{free}}{2} \rceil}^{w} \frac{\tilde{E}(w')T_1(w')T_2(w')}{\begin{pmatrix} N \\ w \end{pmatrix}} \qquad (2)$$

where,

$$T_1(w') = \begin{pmatrix} n\,[2l_{max}(2w) - 1] \\ w' \end{pmatrix}$$

and

$$T_2(w') = \begin{pmatrix} N - n\,[2l_{max}(2w) - 1] \\ w - w' \end{pmatrix}.$$

Hence once the quantities $\{\tilde{E}(w)\}$ have been estimated, the estimate for $E(w)$ can be obtained.

For values of w with $l_{max}(2w)$ close to M, it is generally, more economic in terms of computation to use the technique described in Section 3 to estimate $E(w)$.

5. RESULTS

We demonstrate of the utility of the ETS method described in this paper by evaluating the performance of a rate 1/2 convolutional code with constraint length $L = 6$ and $d_{free} = 8$, given in [6]. The length of the sliding window used by the decoder is $M = 30$. We define types of channel error sequences by looking at the number of channel bit errors in sequences of length 61 time units (or 122 bits).

Consider obtaining the I-BER curve for values of ϵ in the range 0.005 to 0.100. Since the free distance of this code is 8, channel error sequences of types less that 4 will not cause any decoding errors. Hence we need to consider sequences of type $w \geq 4$. For types 4 and 5, the method described in Section 4 was used to estimate $\tilde{E}(w)$. $E(w)$ was calculated using these estimates in (2). The details are tabulated in Table 1. For other values of w, $E(w)$ was estimated using the technique described in Section 3.

Using these estimates of $E(w)$ in equation (1), we obtain the I-BER curve shown in Figure 1. The value of I-BER for some selected values of C-BER are given in Table 2. Table 2 also gives the values of I-BER obtained by the brute-force Monte Carlo simulation and those obtained using the EES technique. It can be seen that our technique produces an estimate that is very close to that produced by extensive Monte-Carlo simulation. This is, however, not surprising, since our technique is in principle equivalent to a brute-force simulation, but is much more efficient. Table 2 also confirms the shortcomings of the EES technique as pointed out in [2], namely, the estimates are biased at moderate to high values of ϵ.

As far as the computational complexity is concerned we present the following analysis.

ETS v/s Monte-Carlo Simulation : Consider obtaining estimates of BER with an accuracy of $\pm 20\%$ about the mean with 95% confidence. If we accumulate over 100 decoding errors for channel error sequences of each type, we are assured of obtaining estimates that lie with in the desired accuracy limits at all values of ϵ. For this ETS required simulation of about $3.7 * 10^6$ bits. Com-

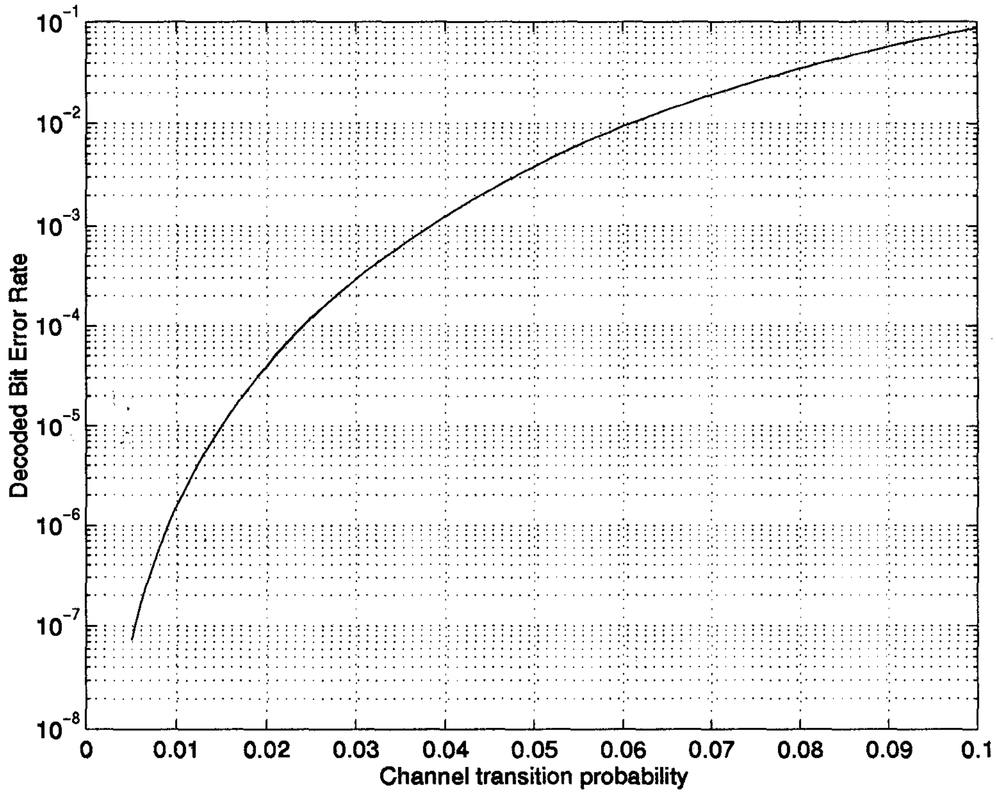

Figure 1: This is the I-BER versus C-BER plot for a rate 1/2 convolutional code with constraint length $L = 6$, see section 5 for details.

pare this with the amount of computation required by the Monte-Carlo simulation to evaluate the code at say $\epsilon = 0.005$. Accumulation of 100 decoding errors would require simulation of about $1.4 * 10^9$ bits. Thus simulation of this *single* channel condition requires about 400 times the computation required by ETS to generate the *entire* I-BER curve.

ETS v/s EES : The EES technique was implemented by simulating error events corresponding to paths of weights 8 – 12. There were 76 such paths in this interval. Simulating 1000 runs for each path with each run requiring decoding of about 25 bits on an average, required a simulation of about $1.9 * 10^6$. Even if we approximate the I-BER curve using, say, 20 points, the computational savings of the ETS method over EES is about 10 times. Thus we see that ETS outperforms the EES method in terms of complexity while maintaining the accuracy of the estimates at *all* channel error rates.

6. CONCLUSION AND DISCUSSIONS

In this paper we presented error type simulation (ETS) - an efficient technique for performance evaluation of convolutional codes. We showed that ETS outperforms the known techniques for performance estimation of convolutional codes in speed as well as in the accuracy of the estimate.

Although the technique was described for the case of hard decision decoding, it should be appreciated that this technique easily generalizes to soft decision decoding on an additive white Gaussian noise (AWGN) channel. For this case we define *types* of channel error sequences by considering the energy of AWGN in a channel output sequence of length $2M+1$ time units. Since this energy is continuous valued, we consider a continuous interval of energy values to define types. With this modification the technique presented in this paper can be used for the performance evaluation of both binary convolutional codes and trellis modulation codes for an AWGN channel. A paper describing these extensions is under preparation.

7. REFERENCES

[1] M. A. Herro and J. M. Nowack, "Simulated Viterbi decoding using importance sampling," *Proc. IEE* , vol. 135, part F, pp. 133-142, Apr. 1988.

[2] J. S. Sadowsky, "A New Method for Viterbi Decoder Simulation Using Importance Sampling,"

IEEE Trans. on Communications , vol. 38, pp. 1341-1351, Sept. 1990.

[3] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding.* New York: McGraw-Hill, 1979.

[4] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications.*, Prentice-Hall, Englewood Cliffs, New Jersey,1983.

[5] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* John Wiley and Sons, 1991.

[6] K. J. Larsen, "Short Convolutional Codes with Maximal Free Distance for Rates 1/2, 1/3 and 1/4," *IEEE Trans. on Info. Theory*, vol. IT-19, pp. 371-72, May 1973.