PERFORMANCE OF BINARY NONSYMMETRIC AND BINARY ERASURE MEMORYLESS CHANNELS WITH VITERBI DECODING USING IMPORTANCE SAMPLING

Bruno B. Albert, Francisco M. de Assis

Departamento de Engenharia Eltrica - Universidade Federal da Paraiba
Caixa Postal 10008 CEP 58109 970, Campina Grande, PB, Brasil {albert,fmarcos}@dee.ufpb.br

Abstract: When estimating the bit error rate (BER) of communications systems using Monte Carlo (MC) simulation, importance sampling (IS) may be a powerful method for reducing simulation runtimes. This paper presents IS technique combined with error event simulation in order to evaluate efficiently the performance of Viterbi decoding used with a binary nonsymmetric channel and with a binary erasure channel. Two biasing aproaches are employed,a stationary and a nonstationary models. Comparative results between the two models and traditional MC are presented.

Keywords: Viterbi Decoding, Error Event, Modified Monte Carlo, Importance Sampling.

1 Introduction

Evaluation of the performance of digital communication system performance, particularly when nonlinearities or band-limited systems are present, is often done via simulation on a computer. The bit error probability P_b is one of the most used parameter for evaluating the digital communication system performance, and a number of simulation techniques can be used to estimate this parameter [1]. The Monte Carlo (MC) method is the most general of the techniques, since no *a priori* assumptions are made. On the other hand, it is computationally the most costly of the methods. This cost is related to the number of observations (samples) that must be made in order to obtain a certain reliability of the estimated parameter. In general, it is required $100/P_b$ samples to obtain a 10% precision related to the estimator's standard deviation of the true value P_b [2]. For very low error probabilities, less than 10^{-6}, computer run times may be prohibitive.

The modified MC technique [3] [4] based on importance sampling (IS) can often reduce significantly the amount of samples for parameter estimating. The procedure is done by "biasing" the noise process in order to give more importance to the relevant events, the error events. By knowing the associated biasing related to each noise sample, it is possible to invert the process and to obtain an unbiased estimator. This technique was successfully applied in nonlinear and non Gaussian channels, particularly in satellite and optical communication channels. However, few works attempt to apply IS to coded systems.

In [2], it is shown that the error event method can be extremely powerful when employed in conjunction with IS. Additionally, it is shown that the efficiency gain does not depend on code constraint length of the convolutional code. In this paper this method is reviewed and applied to a binary memoryless nonsymmetric channel modeling an optical on-off keying (OOK) convolutionally encoded communication system. A binary erasure channel is also modeled. In Section 2 the error event IS simulation is presented. Simulation results are shown in Section 3 and Section 4 concludes the paper.

2 Modified Monte Carlo Method

The modified MC method is based in importance sampling (IS) technique, differently from the conventional MC method in which the occurrence of an relevant event contributes with weight one for the estimate calculation, here, each relevant event is weighted by a value different from one called importance sampling weight. This weighting process is related to the sample statistic distribution changing, in such a way that the relevant (important) events appear more frequently.

2.1 Discrete Memoryless Channel

Given that the input to a discrete memoryless channel (DMC) is a sequence of n symbols $u_0, u_1, ..., u_{n-1}$ selected from the alphabet $\mathcal{X}$ and the corresponding output is a sequence of symbols $v_0, v_1, ..., v_{n-1}$ selected from the alphabet $\mathcal{Y}$, the joint conditional probability is

$$P(Y = v_0, ..., Y = v_{n-1} | X = u_{0,...,} X = u_{n-1}) = \prod_{k=0}^{n-1} P(Y = v_k | X = u_k) \quad (1)$$

This relation mathematically states the memoryless condition of the channel.

2.2 Error Event Simulation Method

Now, we begin a brief review of the Viterbi decoding process fundamentals. More details may be found in [5].

A convenient way to describe the state transitions of a convolutional encoder as a function of time is a trellis diagram. A sequence of connected branches $\mathbf{u} = (u_0, u_1, ...)$ defines a path in the trellis. The code symbols sequence determined by a path $\mathbf{u}$ is defined as $\mathbf{x}(\mathbf{u}) = (x_0, x_1, ...)$, where $x_i = x(u_i)$. The Viterbi decoder output can be viewed as a sequence of correlated decisions on possible branches defined by a trellis path. Decoding errors occur in bursts, called error events, in which a decoded path diverges from the correct path. Formally, an error event $\mathbf{u}^{'}$ is a partial sequence of incorrectly decoded branches, which begins at a correct node, finishes at a correct node, and there is no correct node between them. The number of branches in an error event defines its length, and is denoted by $L(\mathbf{u}^{'})$.

At instant time j, suppose that the encoder state is correctly decoded. For this event, a random variable $N_b(j)$ is defined as the number of erroneously decoded bits due to an error event which begins at time j. Note that from the error event definition, a partial path which begins in a correct node at time j and finishes in a correct node at time $j+1$ is an error event. In this case we call the error event as trivial error event. A trivial error event has 0 wrong bits, $N_b(j) = 0$.

Define the bit error probability when the path $\mathbf{u}$ is transmitted beginning at time j as

$$P_b(\mathbf{u}, j) = E[N_b(j) | \mathbf{x}(\mathbf{u})] \quad (2)$$

where $E[.]$ is the expectation operator. In the case of linear convolutional codes operating on memoryless binary symmetric channel (BSC) and with maximum likelihood decoding, $P_b(\mathbf{u}, j) = P_b$. In other words, the bit error probability does not depend on neither the correct path $\mathbf{u}$ or the time j. In general the bit error probability P_b is defined as

$$P_b = E[P_b(\mathbf{U}, j)] \quad (3)$$

where the expectation is with respect to the random information sequence $\mathbf{U}$. If the channel is stationary but perhaps with memory this expectation is still independent of j.

Consider a linear convolutional code over a memoryless BSC. Without loss of generality, let $\mathbf{u}$ be the all-zero trellis path and $j = 0$, for estimating $P_b = E[P_b(\mathbf{u}, 0)]$.

The error event simulation is based on a sum

$$P_b = \sum_{\mathbf{u} \in \mathcal{E}} n_b(\mathbf{u}, \mathbf{u}') P(\mathbf{u}' | \mathbf{u}) \quad (4)$$

where $\mathcal{E} = \mathcal{E}(\mathbf{u}, 0)$ is the set of all error events which occur at time $j = 0$ when $\mathbf{u}$ is the correct path, $n_b(\mathbf{u}, \mathbf{u}')$ is the number of postdecoding error bits caused by decoding the error event $\mathbf{u}'$ instead of $\mathbf{u}$ when $\mathbf{x}(\mathbf{u})$ is transmitted, and $P(\mathbf{u}' | \mathbf{u})$ is the probability of decoding $\mathbf{u}'$ given the correct path is $\mathbf{u}$ and is called the specific error event probability. Two points must be noted in the above sum. First, the error events $\mathbf{u}' \in \mathcal{E}$ have no fixed length, $L(\mathbf{u}')$ is variable. Second, the sum has infinite terms. However, it is possible to evaluate P_b accurately by concentrating only on dominant terms.

In the error event simulation method, each run considers only one error event of the Viterbi decoder. For understanding the method we have to answer the following question: Given the output channel random sequences $\mathbf{Y} = (\mathbf{Y}_0, \mathbf{Y}_1, ...)$, when a decision is made about the jth trellis branch? This random time instant is denoted by $T_D(j) > j$.

To answer this question, it is helpful to see how the Viterbi algorithm works. Let K be the length of the covolutional encoder shift register. At each time i, 2^K survivor path candidates are recorded, each one terminating at one of the 2^K trellis node. These candidates to survivor branches terminate at distinct nodes then they generate distinct candidate paths. However, if we back trace along these trellis paths, eventually they all merge to a "common stem". As a result $T_D(j)$ is the first time instant that all back trace paths from 2^K candidates merge to a common stem at time j.

The channel output sequences $\mathbf{Y}^{(l)}$ for the lth simulation run is generated by the importance sampling technique, explained in the next subsection.

All candidate paths must begin at the correct node at time $j = 0$. This means that all path are survivor candidates until time $i = K + 1$.

Let $J^{(l)} > 0$ denote the time index of the first decoded branch which merges into a correct node, i. e., the time index at which an error event is decoded. Define $T_M^{(l)} = T_D^{(l)}(J^{(l)})$ as the time at which the lth simulation run detects an error event. Each run only must generate the data sequence $\mathbf{Y}^{(l)}$ until this time. This is an important point: it is not necessary to generate an infinite sequence $\mathbf{Y}^{(l)}$, of course impossible. However, $T_M^{(l)}$ is a random variable, and hence, the simulation has a random length.

2.3 Error Event Simulation and Importance Sampling

When a sequence $\mathbf{x}(\mathbf{u})$ is transmitted, the true conditional joint density of the channel output is $f(\mathbf{y}|\mathbf{x})$. The importance sampling uses a modified joint density, called simulation density, denoted by $f^*(\mathbf{y}|\mathbf{x})$. The importance sampling estimate for the specific error event probability $P(\mathbf{u}'|\mathbf{u})$ is

$$\hat{P}_L^*(\mathbf{u}'|\mathbf{u}) = \frac{1}{L}\sum_{l=1}^{L} w(\mathbf{Y}^{(l)}|\mathbf{x}(\mathbf{u}))\mathbf{I}_{u'}(\mathbf{Y}^{(l)}) \qquad (5)$$

where $\mathbf{I}_{u'}(\mathbf{y})$ is the indicator function for decoding $\mathbf{u}'$, $w(\mathbf{Y}^{(l)}|\mathbf{x}(\mathbf{u}))$ is the importance sampling weight function, which must be defined to make the estimator $\hat{P}_L^*(\mathbf{u}'|\mathbf{u})$ unbiased, and the parameter L is the number of simulation runs for this specific error event $\mathbf{u}'$.

The algorithm stopping time, denoted by $T_M^{(l)} = t$, can be determined by verifying the channel output sequence $y_0,...y_t$, thus $I_{\mathbf{u}'}(.)$ can be decomposed as

$$\mathbf{I}_{\mathbf{u}'}(\mathbf{y}) = \sum_{t=0}^{\infty} \mathbf{I}_{\mathbf{u}',t}(y_0,...,y_t) \qquad (6)$$

where $\mathbf{I}_{\mathbf{u}',t}(y_0,...,y_t)$ is the indicator function of the channel output sequences set $\left\{\mathbf{y} : \mathbf{u}' \text{ is decoded and } T_M^{(l)} = t\right\}$, thus, the equality $\mathbf{I}_{\mathbf{u}'}(\mathbf{y}) = \mathbf{I}_{\mathbf{u}',t}(y_0,...,y_t)$ occurs whenever $\mathbf{y}$ belongs to the above set. If $f_t(y_0,...,y_t|\mathbf{x})$ denotes the true joint density of $(Y_0,...,Y_t)$, and if $f_t^*(y_0,...,y_t|\mathbf{x})$ denotes the importance sampling simulation joint density of $(Y_0,...,Y_t)$, given the sequence $\mathbf{x}$ is transmitted, then the appropriate weight function is

$$w(\mathbf{y}|\mathbf{x}) = \sum_{t=0}^{\infty} w_t(y_0,...,y_t|\mathbf{x})I_t(y_0,...,y_t) \qquad (7)$$

where $I_t(y_0,...,y_t)$ is the indicator function of the channel output set $\left\{\mathbf{y} : T_M^{(l)} = t\right\}$, and $w_t(.)$ is the ratio

$$w_t(y_0,...,y_t|\mathbf{x}) = \frac{f_t(y_0,...,y_t|\mathbf{x})}{f_t^*(y_0,...,y_t|\mathbf{x})}. \qquad (8)$$

The weight defined by the above equation is proved unbiased [2].

3 Simulation Results

We present now some simulation results. We use a convolutional encoder with code rate $R = 1/2$ and constraint length $K = 5$, with generators $g_0 = 23$ and $g_1 = 35$ in octal. This encoder is sufficiently complex to make a MC simulation, in some situations, almost impossible.

We consider first a binary nonsymmetric channel. The following transition probabilities values $P(y_i|x_i)$, see Table 1, are typical of an optical channel with on-off keying (OOK), extremely dispersive [6]

Table 1: Optical channel transition probabilities.

$P(y_i\|x_i)$	$y_i = 0$	$y_i = 1$
$x_i = 0$	$0,9999$	$0,01$
$x_i = 1$	$0,0001$	$0,99$

Two models of error event simulation with importance sampling are compared: a stationary model and a nonstationary model.

The simulation was done in two steps. First, the source transmits only 0's, and second it transmits only 1's. The bit error probability P_b is computed as

$$P_b = P_{b|0}P(0) + P_{b|1}P(1) \qquad (9)$$

where $P_{b|i}$ is the bit error probability when bit i is transmitted, $i = 0, 1$. If the source symbols are generated with equal probabilities than $P_b = (P_{b|0} + P_{b|1})/2$

In the stationary model we used the same biasing value at both steps, $P^*(y_i = 1|x_i = 0) = P^*(y_i = 0|x_i = 1) = 0,08$, when 0's and 1's are transmitted respectively. We used 80.000 samples per error event at each step. These values were defined by trial and error, so as to reduce the estimated relative precision . The estimates of the relative precision are defined by the ratio of the estimator's standard deviation and the estimator mean value. To improve the overall efficiency of the simulation, for the first transmitted encoded symbol through the channel $x_0 = (b_{0,0}, b_{0,1})$, where $b_{i,j}$ denotes the jth bit of the ith transmitted symbol, we use a value 1/2 for the biasing cross probabilities. Formally, this model is not anymore stationary, but we shall continue to call it stationary.

In the Table 2, we present the simulation results for this model. The first column gives the Hamming distance for a given error event $\mathbf{u}'$, and it is the Hamming distance between two symbol sequences $\mathbf{x}(\mathbf{u})$ and $\mathbf{x}(\mathbf{u}')$ until the error event time $L(\mathbf{u}')$. This distance is also called error event distance, and denoted by $d(\mathbf{u}, \mathbf{u}')$. In this simulation we consider error events up to $d(\mathbf{u}, \mathbf{u}') \leq 10$. For a distance d the distance spectrum is defined as the number of error events with Hamming distance d. This parameter is shown in the second column. The information weight is the number of information bit errors $n_b(\mathbf{u}, \mathbf{u}')$ of all error events with $d(\mathbf{u}, \mathbf{u}') = d$, the third column shows these values. The fourth column shows the mean values of the specific error probabilities $P(\mathbf{u}'|\mathbf{u})$ of a specific error event $\mathbf{u}'$ occur given the sequence $\mathbf{x}(\mathbf{u})$ is transmitted, for all error events with $d(\mathbf{u}, \mathbf{u}') = d$. The relative precision is presented percentually in the next column.

In the nonstationary model the biasing value is 1/2 for each bit of the specific error event $\mathbf{u}'$ which differs from the bits of the transmitted path $\mathbf{u}$. In equation form

$$\begin{aligned} P^*(y_i|x_i) &= P(y_i|x_i) \quad && if \;\; b'_{i,j} = b_{i,j} \\ &= 1/2 && if \;\; b'_{i,j} \neq b_{i,j} \end{aligned} \tag{10}$$

where $b'_{i,j}$ denotes a bit of the specific error event $\mathbf{u}'$ which is being evaluated, and $b_{i,j}$ is a bit of the transmitted path $\mathbf{u}$. In the simulation we used 1,000 samples per error event per step. Again, we considered error events up to $d(\mathbf{u}, \mathbf{u}') \leq 10$. The last two columns in the table shows the obtained results for this model.

The estimated bit error probability for the first model is $P_b = 4.320 \times 10^{-9}$ with $2 \times 25 \times 80,000 = 4,000,000$ runs,, whereas for the nonstationary model $P_b = 5.371 \times 10^{-9}$. and takes $2 \times 25 \times 1,000 = 50,000$ runs. A MC model for estimating a BER on the order of 10^{-9} requires a number between 10 to 100 billions runs to obtain a 10% precision. In order to corroborate the above results, we changed the convolutional encoder by another simple one, with $R = 1/2$, $K = 4$, $g_0 = 13$, and $g_1 = 15$. A theoretical analysis showed $P_b < 1.94 \times 10^{-6}$ [6]. We considered now error events up to $d(\mathbf{u}, \mathbf{u}') \leq 8$, and we found $P_b = 2.99 \times 10^{-7}$ with 1.92×10^6 runs for the stationary model (0.08 biasing probability), $P_b = 2.63 \times 10^{-7}$ with 24,000 runs for the nonstationary model, and $P_b = 2.31 \times 10^{-7}$ with 10^9 runs for MC simulation.

The method was also applied to a high level interference channel with a strong signal-to-noise rate. This channel may be modeled by a pure binary eraser channel with no bit errors, in which all interference is detected and blanked [7]. We used the later convolutional encoder in the simulation. If 10% of the total bits are blanked, the estimated BER for the stationary model was $P_b = 1.424 \times 10^{-6}$ with $250,000 \times 12 = 3.0 \times 10^6$ runs (0.2 biasing probability), for the nonstationary model $P_b = 2.058 \times 10^{-6}$ and it takes $1,000 \times 12 = 12,000$ runs. A MC model was also used and we found $P_b = 2.080 \times 10^{-6}$ and 10^8 runs. When 1% of the total bits are blanked, the estimated BER is extremely small, in this case we used only the nonstationary model and we obtained a BER of $P_b = 1.883 \times 10^{-12}$ with $20,000 \times 12 = 240,000$ runs, see Table **??**.

4 Conclusion

IS technique can be effectively applied to a very broad range of systems, but as the system complexity grows, the IS technique must be more finely tuned to the specific problem at hand. The error event simulation method in conjunction with IS proved to be extremely efficient as a tool for evaluate the Viterbi decoder.

The nonstationary model was proved to be faster and more accurate than the stationary model, on the other hand its algorithm is a little more complex. We noted that the error event probability for the stationary model is almost independent of the error event length, this is not the case for the stationary model.

As an extension of this work, we intend to use IS to evaluate digital communication systems which use iterative decoding [8]. A first work was proposed recently in [9], where iteractive decoding is analyzed for product codes using IS. When convolutional codes are used as components codes we may use Viterbi decoders as component decoder, and the method presented in this paper may be a natural base to evaluate these schemes.

Table 2: Comparison of the simulation results for the binary nonsymmetric channel with $K = 5$ convolutional code.

			Stationary Model		Nonstationary Model	
Hamming Distance	Number of Codewords	Information Weight	$P(\mathbf{u}'\|\mathbf{u})$	Relative Precision(%)	$P(\mathbf{u}'\|\mathbf{u})$	Relative Precision(%)
7	2	4	5.01×10^{-10}	26.8%	$1,04 \times 10^{-9}$	3,7%
8	3	12	1.67×10^{-11}	121.4%	$3,36 \times 10^{-11}$	3,9%
9	4	20	1.05×10^{-10}	154.4%	$2,92 \times 10^{-11}$	11,5%
10	20	72	2.58×10^{-13}	330.1%	$5,27 \times 10^{-13}$	8,9%

Table 3: Simulation results for the binary erasure channel when 1% of the total bits are blanked. A $K = 4$ convolutional code was used.

			Nonstationary Model	
Hamming Distance	Number of Codewords	Information Weight	$P(\mathbf{u}'\|\mathbf{u})$	Relative Precision(%)
6	2	4	4.82×10^{-13}	8.53%
8	10	38	5.14×10^{-17}	14.99%

References

[1] Michel C. Jeruchim, "Techniques for estimating the bit error rate in the simulation of digital communication systems," *IEEE Journal on Selected Areas in Communications*, vol. SAC - 2, no. 1, pp. 153 – 170, January 1984.

[2] John S. Sadowsky, "A new method for Viterbi decoder simulation using importance sampling," *IEEE Transactions on Communications*, vol. 38, no. 9, pp. 1341–1351, September 1990.

[3] K. S. Shanmugam and P. Balaban, "A modified Monte Carlo simulation technique for the evaluation of error rate in digital communication systems," *IEEE Transactions on Communications*, vol. COM-28, no. 11, pp. 1916–1924, November 1980.

[4] Dingqing Lu and Kung Yao, "Improved improtance sampling technique for efficient simulation of digital communication systems," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 1, pp. 67–75, January 1988.

[5] G . D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, March 1973.

[6] Stephen B. Wicker, *Error Control Systems for Digital Communication and Storage*, Prentice Hall, Inc., 1995.

[7] Elwyn R. Berlekamp, "The technology of error-correcting codes," *Proceedings of the IEEE*, vol. 68, no. 5, pp. 564–593, May 1980.

[8] Claude Berrou and Alain Glavieux, "Near optimum error correcting correcting coding and decoding: Turbo-codes," *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1261–1271, October 1996.

[9] Marco Ferrari and Sandro Belline, "Importance sampling simulation of turbo product codes," *ICC2001, The IEEE International Conference on Communications*, vol. 9, pp. 2773 – 2777, June 2001.