STUDY & SIMULATION OF MMSE IC-LE TURBO EQUALIZATION SCHEME USING BLOCK (BCH) TURBO CODES

Kanchan Mishra
Bachelor of Technology, Electronics and Communication Engineering
Indian Institute of Technology, Guwahati, India.

Intern: ENST-Bretagne, France (May-July, 2006)

Supervisors:
Ramesh Pyndiah- Head of Signal & Communications Dept., ENST-Bretagne
Dominique Leroux- Associate Professor, ENST-Bretagne

July, 2006.

Contents

Abstract

The fundamental problem of communication, is reproducing at one point , either exactly or approximately,the message selected at another point - C.E Shannon

Turbo equalizers have been shown to be successful in mitigating the effects of Inter Symbol Interference (ISI) introduced by partial response modem and by dispersive channels. In this project, we use a Soft Input Soft Output (SISO) Interference Canceller-Linear Equalizer (IC-LE) based on the Minimum Mean Square Error (MMSE) criterion. The aim of this project was to simulate and study in terms of BER, the performance of MMSE IC-LE SISO equalizers with BCH block turbo codes. We eventually aim to compare in terms of BER, the performance of BPSK receiver with linear turbo equalization using convolutional turbo codes and block turbo codes.

1. Introduction

Digital transmissions over band-limited channels encounter two major impairments to reliable communications, namely additive noise and inter-symbol interference (ISI).Additive noise is a phenomenon of all practical transmission systems and is usually mitigated through the use of proper channel coding techniques (error correcting codes). ISI is the consequence of some form of time dispersion over the channel. Over radio links, this dispersion is generally the consequence of multi-paths propagation while over a telephone channel, it results from imperfect transfer characteristics of the transmission system. ISI arises when successive transmitted symbols are smeared in time and thus overlap at the receiver input up to the point that they may be no longer distinguishable as distinct pulses. From a frequency-domain perspective, the channel transfer function exhibits frequency- dependent attenuations and delays over the transmission bandwidth, hence the alternative name of frequency-selective channels. The impairments caused by ISI are all the more important that successive symbols are spaced more closely together in time in order to increase the data rate, and that the bandwidth restrictions are more stringent. The effects of ISI may be as large on some pathological channels as to preclude reliable communications even in the absence of noise. As an illustration, it is not uncommon today to encounter highly dispersive channels exhibiting delay spreads spanning up to a hundred of symbol periods, as arise for example in the context of underwater acoustic transmissions or broadband wireless access.

Several strategies are known to combat ISI, such as multi-carrier transmissions (OFDM, DMT) or spread-spectrum signaling. We focus in this report on the more classical approach which combines single-carrier transmission with equalization techniques at the receiver. In the presence of channel coding at the transmitter side, we know from classical estimation theory that the optimum receiver should perform equalization and decoding in a jointly manner, by taking simultaneously the constraint imposed both by the code and the ISI channel. The resulting receiver is however intractable in practice and one usually resorts to a sub-optimal two-stages approach where equalization and decoding are performed separately. A major breakthrough occurred in 1995 with the pioneering work of Douillard et al. on Turbo-equalization [1]. This receiver establishes an iterative exchange of probabilistic (“soft”) information between the equalizer and the decoder, so that each function benefits from the result of the other task. It has been observed that at high enough channel signal-to-noise ratios (SNR), the performance of the iterative scheme converges towards the performance of the optimal (joint) receiver while maintaining a significantly lower complexity.

Turbo-equalization is a receiver technique that has evolved over almost a decade of research into a mature technology with promising performance gains over conventional solutions and for which we now foresee practical applications. This report describes the implementation of Minimum Mean-Square Error (MMSE) Interference-Canceller Linear turbo Equalizer (IC-LE) using BCH block turbo codes. We eventually compare in terms of BER, the performance of this MMSE IC-LE equalizer with convolutional turbo and block turbo codes.

1.1 Organization of the report

This report is organized as follows: Section 2 introduces block codes and in particular, BCH codes. Section 3 describes the considered transmission system. Section 4 introduces the turbo-equalization principle and develops the inner equalizer called MMSE IC-LE. Section 5 presents the simulation model wherein we implement the MMSE IC-LE SISO equalizer with BCH block turbo codes. Section 6 compares the performance of the equalizer in terms of BER, with convolutional turbo codes and block turbo codes. Conclusions are finally given in Section 7.

2. Block Codes

Block codes are an important class of error correcting codes that allow correcting errors upto a designated bound. A large number of errors in a codeword can possibly transform a codeword into another valid, but unintended codeword. This situation is called an undetectable error. To guard against this, communication engineers sometimes use an additional overall check code that tests the entire message for validity. Typically, block codes generate horizontal and vertical parity bits for the set of message bits, and finally a single parity bit for the whole code-word block as shown in Fig. 1.

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">n1</td></tr>
<tr><td>n1-k1</td><td>k1</td></tr>
<tr><td rowspan="2">n2</td><td>n2-k2</td><td>Message bits</td><td>Horizontal parity bits</td></tr>
<tr><td>k2</td><td>Vertical parity Bits</td><td>Overall parity bit</td></tr>
</table>

Fig. 1 Structure of a (n1, n2) Block Code

The first linear block code to be developed was the Hamming Code in 1950. However, many of the real world systems use one of the BCH block codes.

2.1 BCH Codes

BCH codes are multiple error correcting, cyclic codes, that form a large class of cyclic error correcting codes, that allow correcting errors up to the designated bound. They were first discovered by A. Hocquenghem in 1959 and independently by R. C. Bose and D. K. Ray-Chaudhuri in 1960. Only the codes, not the decoding algorithms, were discovered by these early writers. The original applications of BCH codes were restricted to binary codes of length 1 2 - *m* for some integer *m*. These were extended later by Gorenstein and Zieler (1961) to the non-binary codes with symbols from Galois field GF(*q*). The first decoding algorithm for binary BCH codes was devised by Peterson in 1960. Since then, Peterson's algorithm has been refined by Berlekamp, Massey, Chien, Forney, and many others.

Primitive BCH codes are given, by the following limits called as the BCH bound. A BCH code exists for a t-error correcting code in $GF(2^m)$, where

$$\begin{aligned} & n = 2t - 1 \\ & n - k \leq mt \\ & d_{min} \geq 2t + 1 \\ & \text{iff, } m \geq 3,\ t \leq 2^{(m-1)} \end{aligned} \qquad (1)$$

2.2 BCH Coding

The generator polynomial for the BCH code is given by $g(x) = LCM\{\varphi_1(x)\varphi_2(x)..... \varphi_{2t}(x)\}$, where φ_r represents the minimal polynomial of α^i, where $O(\alpha) = 2^{(m-1)}$, and α belongs to $GF(2^m)$. BCH being a subclass of cyclic codes, follow the bounds for cyclic codes like

$$c(x) = g(x)m(x) \quad (2)$$
$$v^i(x) = q(x)(x^{(2m-1)} + 1) + v_i(x) \quad (3)$$

representing the codeword $c(x)$ and the cyclic code theorem. This code can correct t or fewer random errors over a span of $(2^m - 1)$ bit positions. Therefore, the code is a t-error-correcting BCH code. Equation (1) describes the process of BCH coding.

The above relations, imply that the generator polynomial has 2t roots, starting from, $a\alpha^2...\alpha^2 t$ and their conjugates. Hence at these values the generator polynomial [and hence the code word],have to have the value of zeros, as every codeword, is a multiple of the generator polynomial. This also means that the minimal polynomials of these 2t values, have to be factors of the code word.

2.3 BCH Decoding

The problem of decoding BCH codes[2] designed to correct t errors, using Bounded Distance Decoding algorithms, is to calculate, up to t errors, and try to correct, them. These lead to a set of key equations, which have upto 2^k solutions, of which we consider the solutions as only those, which help reduce the total number of errors.

Generally BCH decoding itself is organized as follows,
1. Calculate the 2t syndrome for the received word R(x).
2. Find the error-locator polynomial A(x) using any of decoding algorithms, mentioned in this section.
3. Calculate the roots of the error locator, using the Chien's Search algorithm.
4. For non-binary, BCH, we need to use, Forney's algorithm, to calculate the error values, at the positions.

Hence, the (binary) BCH decoding problem just reduces to finding the error locator polynomial, and solving its roots. For this particular problem, several BCH decoding algorithms have been designed, viz. Peterson's Algorithm, Berlekamp Algorithm lead with the Kraft, closed solution to Berlekamp, and Tree decoding or Standard Array, brute-force.

BCH decoders can be of two types: Soft-Input decoder, and Hard-Input decoder. The BCH decoder used in the final simulation model with the equalizer is a soft-input decoder and outputs soft-output and hard-output.

2.4 BCH Codec Performance

As mentioned before, BCH decoders can be of two types: Soft-Input decoder, and Hard-Input decoder. For the coder-decoder given in [Appendix A.1.1, A.1.2], the theoretical bound on bitwise performance, with hard input, is given by-

$$P_b \leq \sum_{i=t+1}^{n} (1+t) C_n^{\,i} p^i (1-p)^{n-i} / n \qquad (4)$$

where, t = number of errors that can be corrected by the BCH decoder,
$p = 0.5 * \mathrm{erfc}(\sqrt{(R_c * E_b/N_0)})$,

and the symbols have their usual meanings.

While studying the performance of MMSE IC-LE SISO equalizer, we shall employ Soft-Input BCH decoder. The theoretical bound on the performance of the decoder based on its bit-wise error correction capability is given by:

$$Pb \leq 0.5 * \sum_{i=1}^{\infty} i/k \sum_{d=d_f}^{\infty} B_{id} * \mathrm{erfc}(\sqrt{(d * R_c * E_b/N_0)}) \qquad (5)$$

where, R_c = Code rate,
$d_f = d_{min}$ for a BCH(n,k,d_{min}) code.

The simulation model for the MMSE IC-LE SISO equalizer uses a BCH(15,7,5) codec. B_{id}'s for BCH(15,7,5) are tabulated below:

Table 1. Bid values for a BCH(15,7,5) codec

$B_{i,d}$ \ d	d_{min}	+1	+2	+3	+4	+5	+6	+7	+8	+9	+10
$B_{1,d}$	5	2	0	0	0	0	0	0	0	0	0
$B_{2,d}$	5	9	4	2	1	0	0	0	0	0	0
$B_{3,d}$	5	13	5	4	5	3	0	0	0	0	0
$B_{4,d}$	3	5	4	5	13	5	0	0	0	0	0
$B_{5,d}$	1	2	5	6	15	6	0	0	0	0	0
$B_{6,d}$	1	2	4	5	13	10	0	0	0	0	0
$B_{7,d}$	1	2	4	5	13	9	0	0	0	0	1

The theoretical performance bounds for BCH(15,7,5) have been obtained using the MATLAB program given in [Appendix A.1.3].

2.5 Results

The simulation of Binary BCH codes, and their performance, for various SNR under AWGN conditions, are discussed in this section. According to Berlekamp the idea of simulating the performance of a code, and assigning it to a particular, communication channel is attributed to Jacobs and Viterbi. This approach finds a suitability of a code, to

particular noise channels, by simulating their performance, under various channel conditions. Now we can choose the best code, for a communication channel, based on the performance of the code, for this channel.

The code for the performance analysis of a BCH(15,7,5) codec is given in [Appendix A.1.4].

2.5.1 Performance of a Hard-input decoder

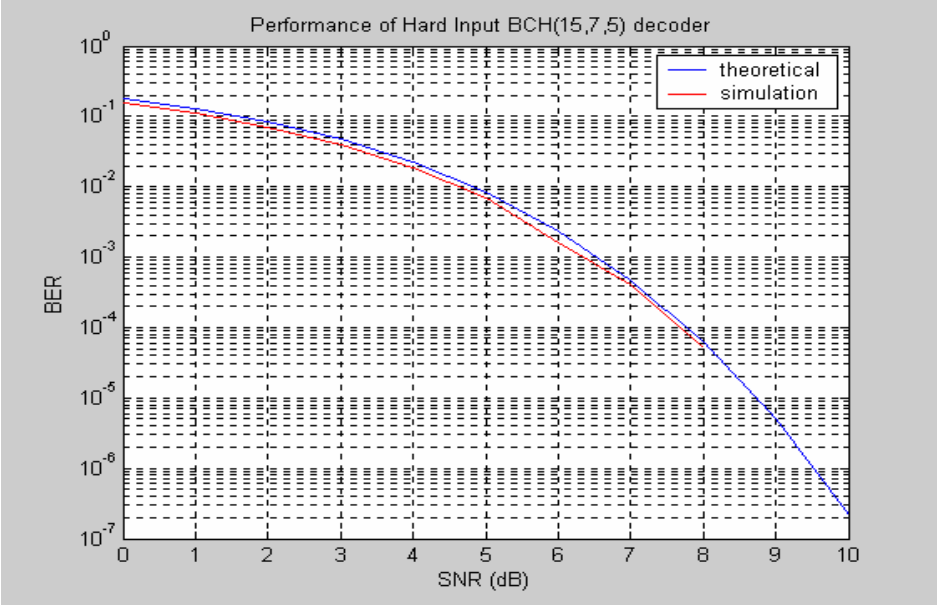

Fig. 2 Performance of a Hard Input BCH(15,7,5) decoder

2.5.2 Performance of a Soft-input decoder

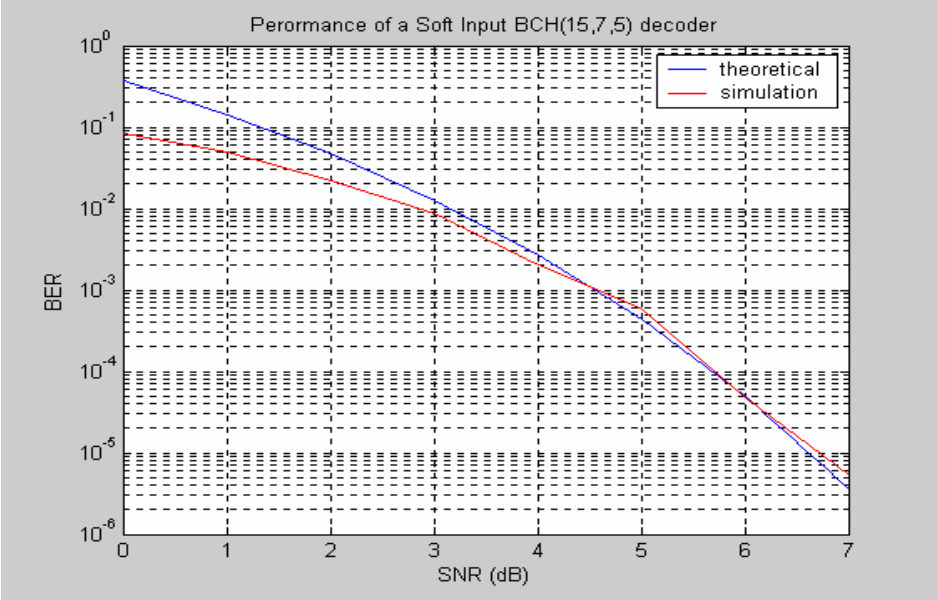

Fig. 3 Performance of a Soft Input BCH(15,7,5) decoder

As seen from the performance curves above, the BER vs E_b/N_0 curve obtained by actual codec simulation closely follows the theoretical performance bound. This proves that the designed codec exhibits good performance over the given AWGN channel.

3. The Transmission System

The block diagram of the communication scheme is shown in Fig. 4. Frames of 4096 information bits are encoded using a rate 7/15 BCH encoder with memory 2 and generator matrix G, given below:

g1= [1 0 0 0 1 0 1 1 1 0 0 0 0 0 0];
g2= [1 1 0 0 1 1 1 0 0 1 0 0 0 0 0];
g3= [0 1 1 0 0 1 1 1 0 0 1 0 0 0 0];
g4= [1 0 1 1 1 0 0 0 0 0 0 1 0 0 0];
g5= [0 1 0 1 1 1 0 0 0 0 0 0 1 0 0];
g6= [0 0 1 0 1 1 1 0 0 0 0 0 0 1 0];
g7= [0 0 0 1 0 1 1 1 0 0 0 0 0 0 1];

G=[g1; g2; g3; g4; g5; g6; g7]; (6)

Tail bits are appended to the coded sequence to ensure zero state trellis termination. The resulting coded bits are interleaved according to a pseudo-random permutation function and mapped onto BPSK symbols with zero mean and unit variance $\sigma_x^2 = 1$. These symbols are transmitted over the channel on a burst-by-burst basis. The transmitted frame consists of a training sequence followed by the payload symbols and is terminated by a guard period of 16 symbols. A coherent receiver front end and perfect synchronization is assumed. The combination of the transmit filter, channel impulse response, receive filter and symbol-rate sampler is represented by an equivalent discrete-time FIR filter with L complex coefficients $\{h_l\}$. Using this notation and transmission scheme, the channel output y_n at time n can be defined as

$$y_n = \sum_{l=0}^{L-1} h_l x_{n-l} + w_n \quad (7)$$

where x_n represents the transmitted symbol at time n and w_n represents the filtered complex Gaussian noise with zero mean and variance σ_w^2.

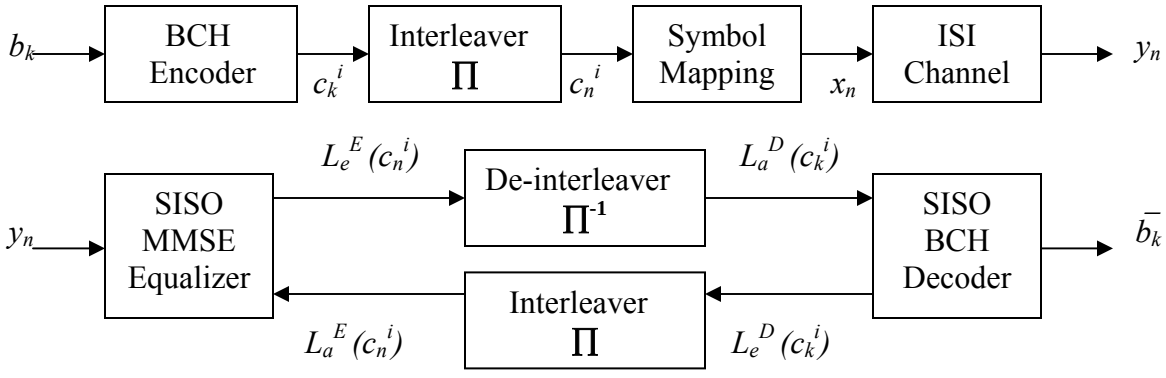

Fig 4. The Communication System

The receiver implements the turbo-equalization scheme shown at the bottom of Figure 4. The turbo-equalizer is made of two Soft-Input Soft-Output (SISO) modules, an equalizer and a channel decoder, separated by interleaving and de-interleaving functions. The SISO equalizer delivers extrinsic information $L_e^E(c_n^i)$ on the coded bits in log-likelihood ratio (LLR) form. The sign of the LLR $L_e^E(c_n^i)$ gives the hard decision (0 or 1) on the coded bit c_n^i while its magnitude measures the reliability of the decision. The extrinsic LLR $L_e^E(c_n^i)$ are de-interleaved and passed to the SISO channel decoder. On the basis of this a priori information, the decoder generates in turn hard decisions $\{b_k\}$ on the information sequence, as well as updated extrinsic information $L_e^D(c_k^i)$ are then interleaved and fed back to the equalizer where they are exploited as a priori information $L_a^E(c_n^i)$ for a new equalization attempt. Iterating the process a few times usually improves significantly the receiver performance. A fixed number of 5 iterations was considered in our application.

The optimal turbo-equalizer relies on a trellis-based SISO equalizer optimized according to the Maximum A Posteriori (MAP) criterion [3]. The complexity of this equalizer however increases exponentially with the number M of symbols in the constellation and the length L of the channel impulse response. This precludes its practical use for high data rate transmissions with multi-level modulation over long delay-spread channels. A vast amount of research efforts has thus been devoted to the design of efficient low-complexity SISO equalizers. Among them, the class of filtering-based SISO equalizers first introduced in [4] offers a competitive alternative. These equalizers maintain a reasonable complexity which grows essentially linearly with the dimension of the signal set and the number of channel taps. This report focuses on a particular filtering-based equalizer proposed in [5] and called MMSE Interference Canceller- Linear Equalizer (IC-LE). Building upon the respective works of [6,7,8], the MMSE IC-LE generalizes the classical MMSE linear equalizer by exploiting the reliability of a priori information available from the turbo-equalization process to adapt the equalization strategy accordingly. The resulting MMSE turbo-equalization scheme appears as an attractive receiver for single-carrier broadband wireless transmissions in severe multi-paths environments.

4. The SISO MMSE IC-LE Equalizer

The structure of the MMSE IC-LE equalizer is depicted in Fig 5. It consists of a soft symbol mapper, interference canceller and SISO demapper. The interference canceller consists of two discrete time FIR filters, the feed-forward filter P(ω) and the feedback filter Q(ω). A complete description as well as the theoretical derivation of the MMSE IC-LE is provided in [5-8]. The main results are stated and used directly in this report.

4.1 Soft Symbol mapping

The soft symbol mapper provides the symbol estimates $\bar{x}_n$. These estimates are calculated as the expected value of the transmitted symbols generated using the apriori information $L_a^E(c_k^i)$ delivered by the decoder. For BPSK modulation scheme, the estimates can be expressed as,

$$\bar{x}_n = \sigma_x * \tanh\left(L_a^E(c_n^1)/2\right) \tag{8}$$

For QPSK modulation scheme, the estimates can be expressed as,

$$\bar{x}_n = (\sigma_x/\sqrt{2})*[\tanh\left(L_a^E(c_n^1)/2\right) + j*\tanh\left(L_a^E(c_n^2)/2\right)] \tag{9}$$

where the LLRs $L_a^E(c_n^1)$ and $L_a^E(c_n^2)$ correspond to the symbol x_n.

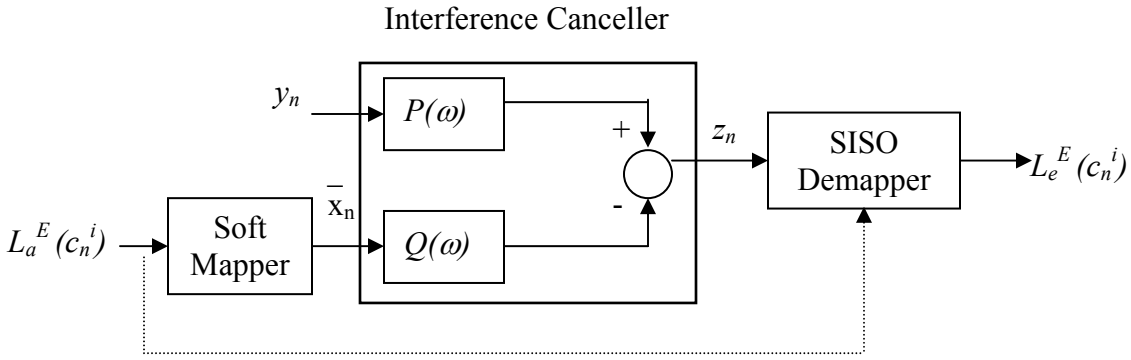

Fig 5. The SISO MMSE IC-LE

Additionally, the soft symbol mapper also computes the estimated symbol variance σ_x^2. With $E(\bar{x}_n) = 0$, the estimated symbol variance becomes,

$$\sigma_x^2 = E(|\bar{x}_n|^2) \approx (1/N)\sum_{n=0}^{N-1} |\bar{x}_n|^2 \tag{10}$$

Initially, there is no apriori information available and $\Pr(c_n^i=0) = \Pr(c_n^i=1) = 1/2$. Hence, we have $L_a^E(c_n^i)=0$, estimated $\bar{x}_n = 0$ and $\sigma_x^2 = 0$. As the reliability of the LLR's increase with the iterations, $\bar{x}_n \to x_n$ and $\sigma_x^2 \to \sigma_x^2$.

4.2 Interference Canceller

As indicated in Fig. 5, the equalized sample at time n is given by,

$$z_n = \sum_l p_l y_{n-l} - \sum_m q_m \bar{x}_{n-m} \tag{11}$$

where the reference tap of the feedback filter q0 is set to zero to prevent the cancellation of the desired signal. The filter coefficients are obtained according to the minimization of the mean-square error $E(|z_n - x_n|^2)$. The coefficients are computed once a burst from an estimate of the channel impulse response (CIR) and applied to the entire received sequence. The CIR estimate is typically obtained from a known training sequence embedded in each transmitted packet. With $H(\omega)$ being the Fourier transform of the CIR, we get,

$$P(\omega) = \frac{\sigma_x^2}{1+\beta\sigma_x^2} \frac{H^*(\omega)}{(\sigma_x^2 - \sigma_x^2)|H(\omega)|^2 + \sigma_w^2} \tag{12}$$

where $$\beta = \frac{1}{2\pi} \int_{-\pi}^{+\pi} \frac{\sigma_x^2 |H(\omega)|^2}{(\sigma_x^2 - \sigma_x^2)|H(\omega)|^2 + \sigma_w^2} d\omega$$

and, with $G(\omega) = H(\omega)P(\omega)$,

$$Q(\omega) = G(\omega) - g_0, \text{ with } g_0 = (1/2\pi) \int_{-\pi}^{+\pi} G(\omega)\, d\omega \tag{13}$$

Table 2. Filters coefficients computation procedure

1. Compute the FFT $\{H_n\}$ of $\{h_n\}$ on N_p points.
2. Compute $D_n = (\sigma_x^2 - \sigma_x^2)|H_n|^2 + \sigma_w^2$ and $P_n' = H_n^* / D_n$ for n=0,….,N_p-1.
3. Compute $\beta = (1/N_p) \sum_{n=0}^{Np-1} H_n P_n'$ and $g_0 = \sigma_x^2 \beta / (1+\beta\sigma_x^2)$.
4. Compute $p_n = \sigma_x^2 P_n' / (1+\beta\sigma_x^2)$ and take the IFFT of $\{P_n\}$ on N_p points to get the time-domain impulse response $\{p_n\}$.
5. Compute $\{q_n\}$ as the convolution of $\{p_n\}$ and $\{h_n\}$ and set $q_0 = 0$.

$P(\omega)$ and $Q(\omega)$ theoretically have infinite lengths and thus cannot be implemented practically. The alternate finite-length solution using matrix algebra involves inversions with $O(N_p^2)$ complexity, N_p being the number of taps of the feed forward filter $P(\omega)$.Instead, an approximate and low-complexity alternative relying on the Fast Fourier transform (FFT) with complexity $O(N_p \log_2 N_p)$ is used to determine the filter coefficients. The resulting procedure is summarized in Table 2. the feedback filter $Q(\omega)$ has length $N_q = N_p + L - 1$.

4.3 SISO symbol demapper

The SISO demapper computes the extrinsic LLR $L_e^E(c_n^i)$ on the coded bits, from the knowledge of the received sample and apriori LLR $L_a^E(c_n^i)$ for higher-order modulations. The equalized sample can be written as,

$$z_n = g_0 x_n + v_n \tag{14}$$

where v_n denotes the residual noise and interference at the equalizer output. Assuming that v_n is Gaussian with variance σ_v^2, it can be shown that,

$$\sigma_v^2 = \sigma_x^2 g_0(1-g_0) \tag{15}$$

$L_e^E(c_n^i)$can thus be expressed in the QPSK case as,

$$L_e^E(c_n^1) = \frac{4}{\sqrt{2}(1-g_0)} Re(z_n) \tag{16}$$

$$L_e^E(c_n^2) = \frac{4}{\sqrt{2}(1-g_0)} Im(z_n) \tag{17}$$

5. Simulation of SISO MMSE IC-LE using BCH Block Turbo Codes

The simulation of binary BCH codes, and their performance for various SNR under AWGN conditions, were discussed in Section 2. Section 3 explained how the transmission system model has been approximated. Section 4 discussed the system model used for SISO MMSE IC-LE. In this section we implement the SISO MMSE IC-LE with BCH(15,7,5) codec and simulate the system to study the performance of SISO MMSE IC-LE using Block turbo codes [9].

5.1 Simulation as a Method

The idea of simulating the SISO MMSE IC-LE with the BCH codec is to study the performance of linear MMSE turbo-equalizers with block turbo codes. This approach finds the suitability of block turbo codes to particular noise channels, by simulating their performance, under various channel conditions. Further, comparison of the simulated performance of MMSE turbo-equalizers with convolutional turbo codes and block turbo codes[10], gives us a fair idea of their performance in practical scenarios. Now we can choose the best code, for a communication channel, based on the performance of the code, for this channel.

5.2 Simulation Model

The system model has been shown in Fig. 4. The various blocks have been modeled as followed:

1. Source

We take the random bits 1's and 0's so that we simulate a ergodic source that is input to the channel coder. The random bits are generated using the MATLAB function rand( ). The source has a memory of 2.

2. BCH Encoder

BCH encoder takes as input the data generated by the source and the data block size. The generator matrix given in (6) is used with the data matrix to obtain the coded matrix using equation (2) in discrete form. To obtain the coded data in binary form, we take the mod of 2 for the matrix thus obtained. Finally, the coded bits are transformed from unipolar {0s and 1s} to bipolar state {-1s and 1s}. [Ref.: Appendix A.2.2].

3. Interleaver/ Deinterleaver

The interleaving (deinterleaving) function prevents the occurrence of block errors due to channel noise. The coded bits resulting from 2, are interleaved according to a pseudo-random permutation matrix p generated by the pseudo-random permutation function that

uses a user defined matrix of same size as the data matrix, to create the interleaver/ deinterleaver matrix. On the deinterleaver side, the deinterleaver matrix is used to revert the effect of interleaving. [Ref.: Appendix A.2.3].

4. Symbol mapping / Modulation

The soft symbol mapper provides the symbol estimates $\bar{x}_n$. It takes as input, the interleaved coded matrix and the type of modulation to be performed and outputs the symbol estimates. [Ref.: Appendix A.2.4].

5. Channel

The code has provision for channel type selection. System model allows Porat, Macchi, ProakisA, ProakisB, ProakisC, Crit3, Crit4, Crit5, Crit6 and 5-path Gaussian channel models to be analyzed. During passage through the channel, the symbols get convolved with the channel impulse response and additive noise is added to the resulting convolution output (7). Noise is a random sequence of bits generated using the function randn( ) and having an amplitude in accordance with the desired SNR.

6. Interference Canceller

The interference canceller implements the feed-forward filter $P(\omega)$ and the feedback filter $Q(\omega)$ shown in Fig 5. The filter coefficients are computed according to the steps given in Table 2. The output of the interference canceller is then computed using (11). [Ref.: Appendix A.2.5].

6. Demapping

The SISO demapper computes the extrinsic LLR on the coded bits, from the knowledge of the received sample and apriori LLR $L_a^E(c_n^i)$ for higher-order modulations using (16) and (17). [Ref.: Appendix A.2.6].

7. BCH Decoder

The decoder takes soft input and gives both soft and hard decisions as outputs. It also takes as an input parameter the variance of the additive channel noise for computing the soft decision. [Ref.: Appendix A.2.7].

8. Destination

This is the information sink, that accepts the decoded bits from the channel encoder, and extracts the message, bits from the codeword, and reproduces the data at the receiver end, reliably.

The Source, Channel and Destination have been implemented as part of the main program. [Ref.: Appendix A.2.1].

The communication model described above is used, and inputs are a fixed number of random bits, to be encoded, and passed into the system. Next the same simulation is used for various values of SNR against which the plots are to be drawn. The SNR values in dB are converted to obtain the signal power, assuming a constant Noise power, which provides for our uniform random noise AWGN, with a fixed power [given as variance]. This value of signal power is used, to calculate the input signal energy, to the BPSK signal modulator, and to the signal demodulator.

Now for each experiment, we have to calculate the number of errors between the output of the encoder at the Tx end, and the output of the decoder, at the Rx end. This basically means that, you can calculate the BER by dividing the total number of errors for this SNR, by the total number of bits transmitted. BER = n(errors)/ n(bitsTx).

For each experiment run, at a particular SNR, we will have a specific, BER. Using this we may to calculate, the plot the SNR vs BER curves for the coded system we have used.

5.3 Factors affecting the Simulation

We can easily appreciate the fact, that simulations of coding systems, need to have large 10^6 number of bits that are simulated in the system. This would ensure us, a high accuracy and enable us to achieve BER rates of the order of 10^{-5}, or so. For any system to achieve a BER of 1/R we need to transmit at least 10*R bits, across the channel. This order of a magnitude difference, in the transmitted bits, increases the confidence in our simulation results and it is in perfect agreement with the law of large numbers [11].

We can also appreciate the errors that may creep up in interpreting, the simulations; if we don't have a BER, i.e. BER = 0 then, it means one of the following:
1. BER is actually 0.
2. The number of bits, transmitted across the channel, is very less.
3. SNR is way too high or, noise floor, is way too low.

A second problem encountered is the speed of running of the program. Due to iterative nature of the decoder, its implementation on MATLAB is very slow due to nested loopings. This may result in degradation in system performance. Thus, a decoder written in C/C++ is recommended. It can be linked to the main program in MATLAB using the mex function. This improves the decoder speed and system performance. [Ref.: Appendix A.4].

5.4 Results

5.4.1 Genie Iteration

Genie iteration is the name for perfect estimation scenario i.e. when the input to the feedback filter are the actual sent symbols rather than the estimated symbols. This results in the best output being obtained in the first iteration itself and is the ideal case.

The simulation for obtaining the genie iteration curve utilizes the program for IC-lE given in Appendix A.2.4(a) and the main program given in Appendix A.2.7(a).

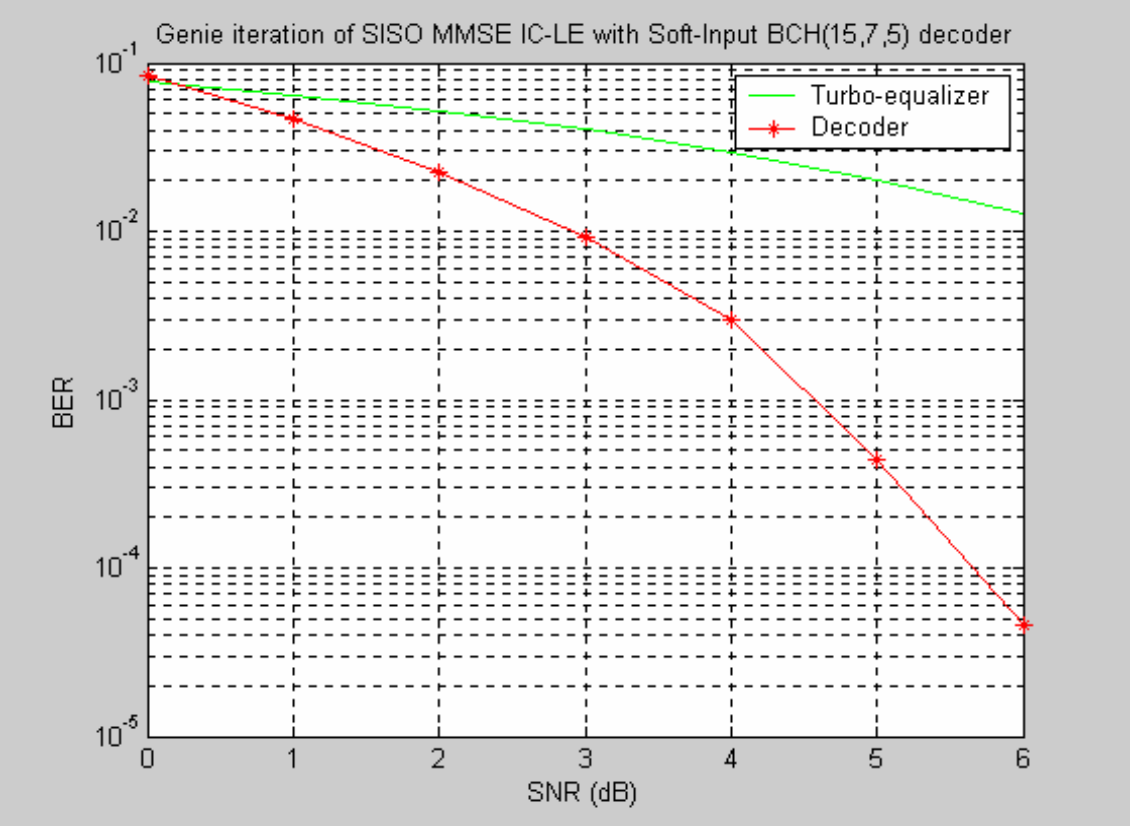

Fig. 6 Genie iteration of SISO MMSE IC-LE with soft-input BCH(15,7,5) decoder (Porat Channel)

5.4.2 Iterative Performance

In the iterative performance analysis, the input to the feedback filter with impulse response $Q(\omega)$ are the estimated symbols. With increasing number of iterations, the performance curve should tend to the Genie iteration curve.

The simulation for analyzing the iterative performance utilizes the program for IC-lE given in Appendix A.2.4(b) and the main program given in Appendix A.2.7(b). The results obtained for four iterations are plotted in Fig. 7.

The numbers along each curve represent the iteration number. The blue curve corresponds to the performance of the IC-LE in terms of BER whereas the green curve corresponds to the performance of the decoder in terms of BER. As can be seen from the figure, the performance of the linear equalizer and the decoder improve with the number of iterations.

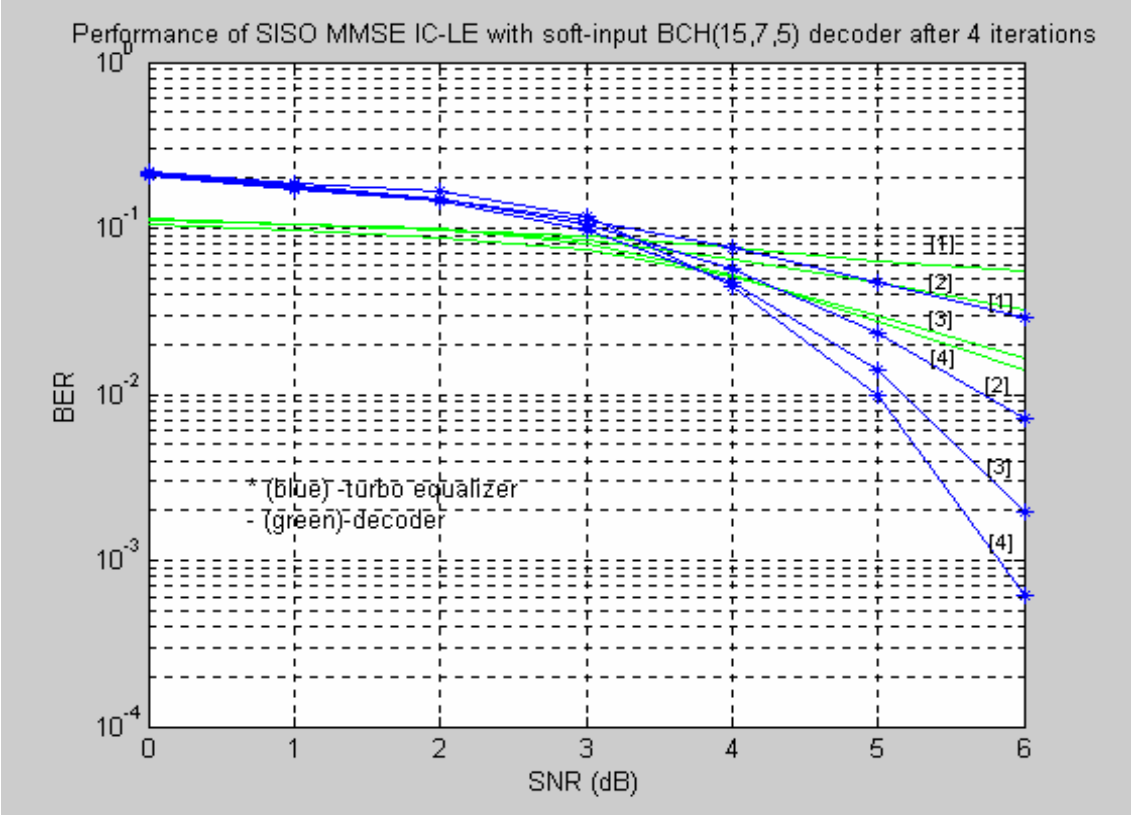

Fig. 7 Performance of SISO MMSE IC-LE with soft-input BCH(15,7,5) decoder after 4 iterations (Porat Channel)

6. Performance of SISO MMSE IC-LE using Convolutional turbo codes

The program provides the users with the provision of analyzing the performance of SISO MMSE IC-LE using Convolutional turbo codes for various channels and compare its performance using block turbo codes and convolutional turbo codes.

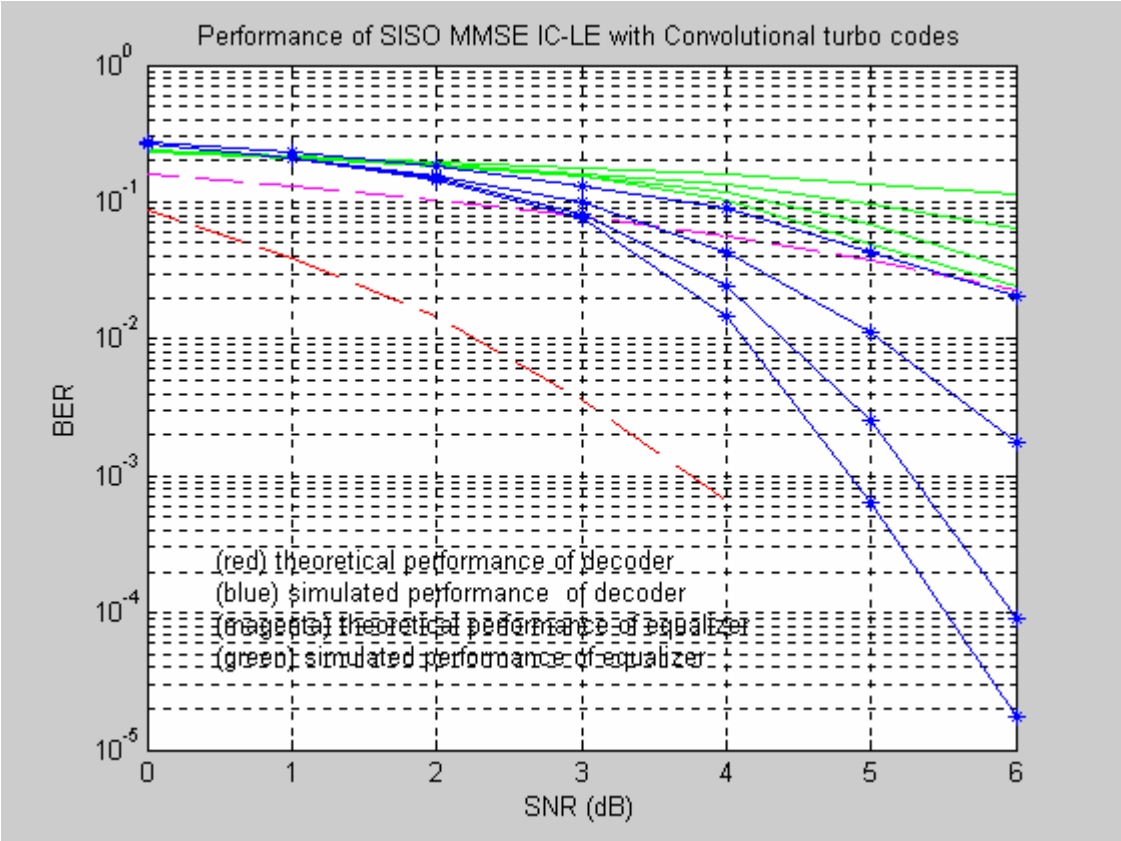

Fig. 8 Performance of SISO MMSE IC-LE with Convolutional turbo codes after 4 iterations (Porat Channel)

7. Conclusions

The turbo equalization performance of BPSK modulated transmission systems using BCH turbo codes, and convolutional turbo codes was compared at code rates of ½ (convolutional codes), and 7/15 (BCH block codes). Our comparative study of the turbo equalizers showed that for the Porat channeland also other typical channels, and at high code rates, the convolutional turbo coded system CT yielded better performance, when compared with the BCH turbo coded BT system.

When compared in terms of number of iterations required to obtain near ideal results, CT systems again outperform BT systems as BT systems need more iterations to obtain the near ideal performance. The CT system incurs moderate computational complexity compared with the BT system. BT systems have higher decoder complexity resulting in slow decoding speed.

However, simulation of the system with BCH block turbo codes under various noise channels gives us an idea of the suitability of this system with various noise channels and in some cases BT systems certainly hold the cutting edge. It also enables us to choose the best code, for a communication channel, based on the performance of the code, for this channel.

B. REFERENCES

[1] C. Douillard, A. Picart, P. Didier, M. Jezequel, C. Berrou and A. Glavieux, "Iterative correction of intersymbol interference: Turbo- Equalization," *European Trans. Telecommunication*, vol.6, no.5, Sept./Oct. 1995.

[2] J. Hagenauer, E. Offer and L. Papke, " Iterative Decoding of Binary Block and Convolutional Codes," *IEEE Trans. on Infor. Theory,* Vo1.42, No.2, pp.429- 445, Mar. 1996.

[3] G. Bauch, H. Khorram and J. Haganauer, " Iterative Equalization and Decoding in Mobile Communication Systems," *Proc. 2nd European Personal Mobile Commun. Conf. EPMCC'97*, Bonn, Germany, Sept.- Oct. 1997, pp. 307-312.

[4] A. Glavieux, C. Laot and J. Labat, "Turbo- Equalization over a Frequency selective Channel," *Proc. 1st Int. Symp. on Turbo-Codes & Related Topics,* Brest, France, 3-5 Sept. 1997, pp. 96-102.

[5] M.Tuchler, A.C. Singer and R. Koetter, " Minimum Mean- Square Error Equalization with priors," *IEEE Trans. Signal Processing,* vol.50, no.3, pp. 673-683, Mar. 2002.

[6] C. Laot, A. Glavieux, and J. Labat, "Turbo Equalization: adaptive equalization and channel decoding jointly optimized." *IEEE J.Sel. Areas in Commun.*, vol. 19, no. 9, pp. 1744-1752, Sep.2001.

[7] R. Le Bidan, "Turbo- Equalization for bandwidth-efficient digital communications over frequency- selective channels," Ph.D. dissertation, INSA de Rennes, Rennes, France, Nov.2003.

[8] R. Koetter, A. C. Singer and M. Tuchler, " Turbo - Equalization." *IEEE Signal Process. Mag.* Vol.21, no.1, pp. 67-81, Jan.2004.

[9] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes." In *Proc. Int. conf. Communications*, Geneva, Switzerland, May 23-26, 1993, pp. 1064-1070.

[10] B.L.Yeap, T.H. Liew, J. Hamorsky and L. Hanzo, " Comparative Study of Turbo Equalization Schemes using Convolutional, Convolutional Turbo, and Block-Turbo Codes."

[11] Eric W. Weisstein et al. "Law of Large Numbers." From Math World–A Wolfram Web Resource. http://mathworld.wolfram.com/LawofLargeNumbers.html