SWITCHING METHODS FOR LINEAR TURBO EQUALIZATION

Seok-Jun Lee, Naresh R. Shanbhag, and Andrew C. Singer

Coordinated Science Laboratory, Electrical and Computer Eng. Dept.
University of Illinois at Urbana-Champaign
1308 West Main Street, Urbana, IL 61801
Email: [slee6,shanbhag,acsinger]@uiuc.edu

ABSTRACT

In this paper, several switching methods are presented for a class of switching turbo equalization, where the performance improvement is achieved via adapting the choice of equalizers during the iterative procedure. Four techniques for equalizer selection are presented and compared in computer simulations. The switching scheme based on the average of soft information provided by SISO decoders showed the best bit error rate (1.5dB processing gain at 10^{-4}) with low complexity.

1. INTRODUCTION

The turbo principle first demonstrated in turbo code decoding [1] has been applied to a variety of communication systems. As a result, *turbo equalizer* was proposed [2] to protect data transmission over an intersymbol interference (ISI) channel. The original turbo equalization technique [2] employed maximum *a posteriori* probability (MAP) equalizer and decoder, but has impractically high complexity [3]. Thus, a class of low-complexity soft-input soft-output (SISO) equalizers were proposed [3]–[5] by replacing the MAP equalizer with a linear SISO equalizer, resulting in the so-called "linear turbo equalizer". Linear turbo equalizers enable substantial performance gains over systems in which equalization and decoding are performed separately (6-10 dB processing gain after five iterations for typical channels) [3]–[5].

Switching turbo equalizers select an equalizer from a family of equalizers in each iteration. This results in superior receiver performance [6]. However, the method for equalizer selection has not been addressed so far. In this paper, four techniques are presented for SISO equalizer selection. These techniques are compared in simulations and their complexities are evaluated.

This material is based upon work supported by the National Science Foundation under Grant CCR 99-79381, CCR-0092598, and ITR 00-85929.

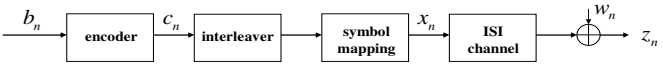

Fig. 1. Channel model including the transmitter.

2. SWITCHING LINEAR TURBO EQUALIZER

In this section, we briefly describe the switching linear turbo equalizer architecture and the computation of these equalizer coefficients.

2.1. Architecture

The system model we assume has a transmitter as depicted in Fig. 1 with block-based transmission. Binary data b_n is encoded yielding the coded sequence c_n, which is then permuted by the interleaver. The interleaved sequence is mapped onto symbols $x_n \in \{-1, +1\}$ in case of BPSK modulation, which are transmitted over an ISI channel with additive white Gaussian noise (AWGN).

Figure 2 depicts a typical switching linear turbo equalizer architecture, where more than two sets of coefficients can be employed. In this paper two sets of equalizer coefficients are selected according to the current state of the iterative procedure. The linear SISO equalizer consists of two operations: symbol estimation and soft-information mapping of estimated symbols. The overall architecture of a SISO equalizer is similar to that of the well-known decision-feedback equalizer (DFE). However, instead of hard decisions (quantized), soft symbols $\bar{x}_n$ are passed to a feedback filter from the previous iteration of the SISO decoder. The log-likelihood ratio (LLR) mapping converts each estimated symbol $\hat{x}_n$ to a LLR $L_o^E(\cdot)$, which is calculated as

$$L_o^E(x_n) = \ln \frac{Pr(\hat{x}_n | x_n = +1)}{Pr(\hat{x}_n | x_n = -1)}, \qquad (1)$$

for BPSK signals. The superscript E and D denote equalization and decoding, respectively, and the subscript o and i refer to the output and input, respectively, in Fig. 2.

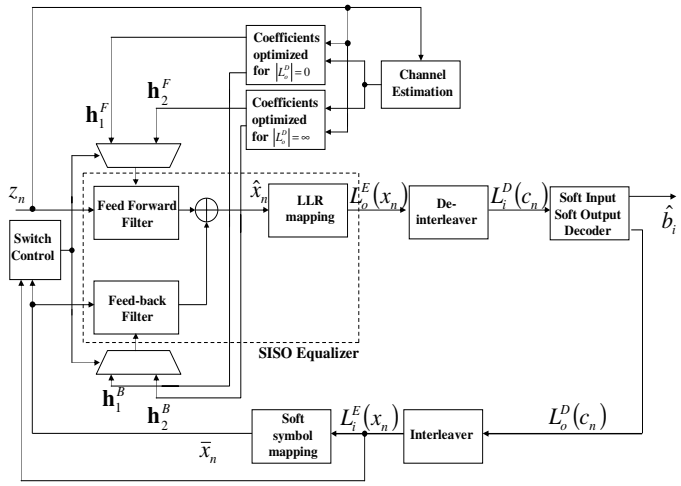

Fig. 2. A switching linear turbo-equalizer block diagram.

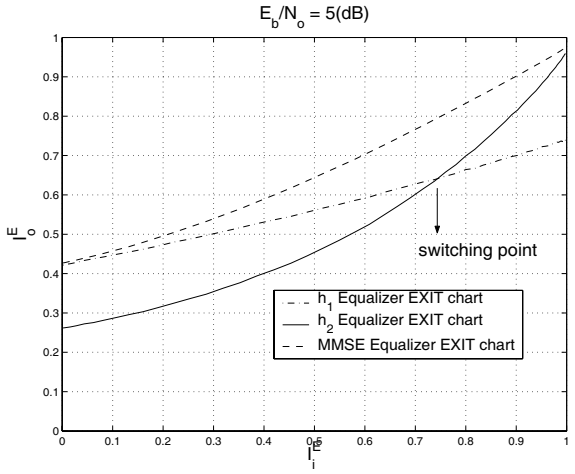

Fig. 3. Two EXIT charts of SISO equalizers with $\mathbf{h}_1$ and $\mathbf{h}_2$.

The updated $L_o^E(\cdot)$ are fed to the SISO decoder after de-interleaving, and the decoder attempts to improve the soft information on the coded bits, c_n, and produces $L_o^D(\cdot)$, the LLR of each coded bit. In turn, $L_o^D(\cdot)$ is passed to a soft symbol mapping block and then converted to the soft symbol, $\bar{x}_n$, which is computed as

$$\bar{x}_n = \frac{\exp(L_i^E(x_n)) - 1}{1 + \exp(L_i^E(x_n))}, \tag{2}$$

where $L_i^E(x_n)$ is the interleaved version of $L_o^D(c_n)$ [3]. This soft symbol is then fed back to the equalizer block for the next iteration. The details of such a SISO decoder algorithm are described in [7].

2.2. Coefficient computation

Given the current set of decoder soft information, the best (in a minimum mean squared error (MMSE) sense) linear SISO equalizer must re-compute the coefficients for estimating each symbol [5]. This is computationally complex. The switching linear turbo equalizer avoids this complexity by employing two sets of coefficients which are optimized for less and highly reliable soft symbols. In [4], these two sets of linear equalizer coefficients were computed in an MMSE sense using perfect channel knowledge. The first set $\mathbf{h}_1$ is computed assuming that no LLR information is provided by the SISO decoder, i. e., with $L_o^D = 0$. The second set $\mathbf{h}_2$ is computed assuming perfect LLR information, i.e., with $|L_o^D| \to \infty$. Thus, the switching turbo equalizer employs $\mathbf{h}_1$ for the first few iterations when L_o^D is indeed close to 0 and then switches to $\mathbf{h}_2$ after $|L_o^D|$ becomes large enough. These two sets of coefficients can be computed during a training period by estimating the channel information.

3. EXIT CHART ANALYSIS

A convenient tool in visualizing the soft information evolution characteristics of linear SISO equalizers is the extrinsic information transfer (EXIT) chart [4, 8]. The EXIT chart traces one single parameter to observe the soft information evolution. This parameter is the mutual information I_i^E or I_o^E ranging from 0 (no knowledge of the transmitted symbols) to 1 (transmitted symbols are perfectly known). Here, I_i^E and I_o^E are the mutual information between soft information (L_i^E or L_o^E) and x_n (transmitted symbols), respectively. To obtain the linear SISO equalizer EXIT chart, the input soft information L_i^E is reasonably well approximated [4, 8] as independent and identically distributed with

$$f_L(l|x_n) \triangleq f_L(l|X = x_n) = N(\frac{x_n \sigma_L^2}{2}, \sigma_L^2), \tag{3}$$

where σ_L^2 is the variance of soft information. Then, given each L_i^E distribution (I_i^E), the output mutual information I_o^E is measured and plotted in Fig. 3, where the switching point (crossing point of two EXIT charts) is clearly observed. Thus, the soft information evolution trajectory of a switching turbo equalizer should at first follow the EXIT chart of the $\mathbf{h}_1$ equalizer in the less reliable region and then switch to the $\mathbf{h}_2$ equalizer after the switching point. Doing so enables the performance of switching turbo equalization to approach that of the MMSE equalization more closely.

However, it is very difficult to measure the mutual information in practice to determine the switching point. In the next section, we presents four switching methods which can be implemented in VLSI circuits with lower complexity than computing the mutual information I_i^E directly.

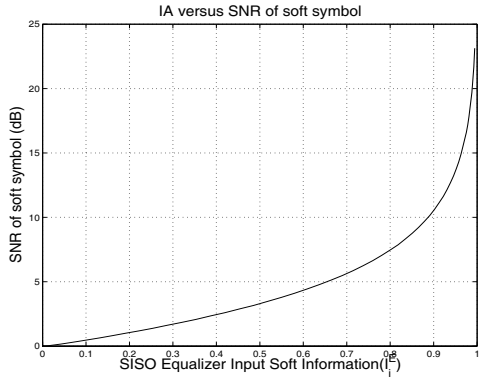

Fig. 4. η(SNR of soft symbols $\bar{x}_n$) over the SISO equalizer input mutual information.

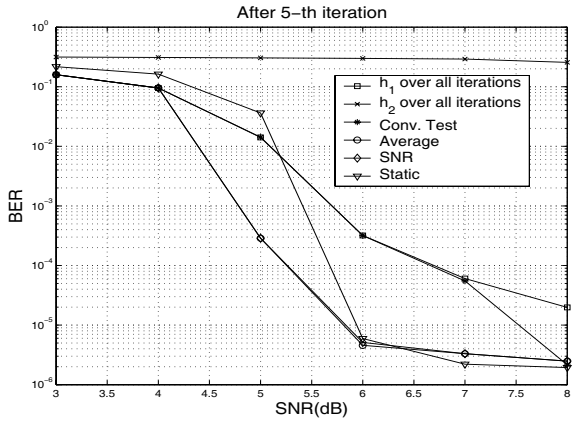

Fig. 5. Comparison of switching methods in terms of BER after 5- iteration.

4. SWITCHING METHOD

Switching turbo equalization enhances the receiver performance at the marginal cost of one additional set of coefficients. To enable the performance enhancement in VLSI implementation, the switching point should be determined with low complexity in the receiver. Further, since each finite-sized block can have slightly different convergence rates [4], the switching point may be different from each received block. Thus, it is necessary to determine a switching point depending on the current status of iterative procedure. In this section, four switching schemes are presented.

4.1. Static switching method

One simple solution would be to employ a static switching algorithm. The switch control logic in Fig. 2 simply compares the pre-set switching point (S_1) with the number of executed iterations (S_2) and if $S_2 > S_1$, then the switching occurs. Thus, the switch control logic requires one counter and one comparator.

4.2. SNR-based method

If we define the SNR of soft symbols $\bar{x}_n$ as

$$\eta = 10 \log_{10} \frac{x_n^2}{(x_n - \bar{x}_n)^2}. \quad (4)$$

The SNR η becomes large as the soft symbols become more reliable (see Eq. (2)) and is a function of the SISO equalizer input mutual information. Figure 4 shows the evolution of η versus the SISO equalizer input mutual information. The switching point S in Fig. 3 can be represented in terms of η. From EXIT chart analysis, $I_i^E = 0.75$ is determined as a switching point. Hence, an equivalent SNR value is set to 7 dB (see Fig. 4). The switch control logic in Fig. 2 compares the current iteration η with the pre-set threshold value η_T and, if $\eta \geq \eta_T$, then the equalizer is switched. The switch control logic needs one accumulator for $(x_n - \bar{x}_n)^2$ and some extra hardware to implement the square and log functions.

4.3. Average-based method

Another way to avoid the computation of I_i^E is to compute the average of the absolute values of L_i^E over one data block. As the soft information gets reliable, it is known that the absolute values of L_i^E become larger [4], [8]. Since I_i^E is also a function of L_i^E, the average of absolute values of L_i^E can determine the switching point instead of I_i^E. If the average is greater than a pre-set threshold, equalizers are switched. The pre-set threshold can be determined empirically. The complexity is much reduced in comparison with SNR-based method since the switch control logic needs only one accumulator and one comparator.

4.4. Convergence test method

Switching from the first equalizer to the second occurs when the first equalizer converges. In order to detect the convergence, the decoded bits of the previous iteration and those of the current iteration are compared. If the number of mismatched decisions is less than a threshold (which can be determined empirically), the control logic switches to the second equalizer coefficients. The switch control logic is simple, but needs extra storage to hold the decisions from the previous iteration.

5. EXPERIMENTAL RESULTS

In this section, the proposed switching schemes are compared in computer simulations.

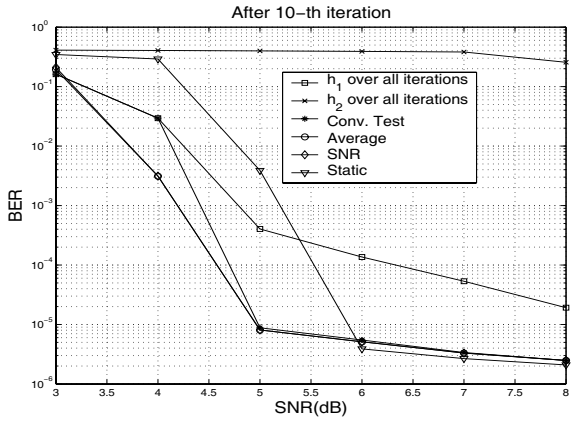

Fig. 6. Comparison of switching methods in terms of BER after 10- iteration.

5.1. Simulation setup

We employ a recursive systematic convolutional (RSC) encoder at the transmitter with a generator polynomial $(23, 35)_8$. The coded bit stream is first passed through a random interleaver followed by a modulator. Here, we consider BPSK and the channel model

$$\mathbf{H}(z) = 0.227z^2 + 0.46z + 0.688 + 0.46z^{-1} + 0.227z^{-2}, \quad (5)$$

where $\mathbf{H}(z)$ has severe ISI (strong spectral null near $\omega = 0.6\pi$). We use 65536-bit random interleavers. The number of taps used in the feedforward path is 11 and the number of taps in the feedback path is 15. A sliding window Log-MAP decoder [7] is employed and 10 iterations are carried out. The static switching method switches the coefficients in the 3rd iteration. To determine the switching point, the SNR-based method chooses 7 dB as a threshold value and the average method compares the mean of absolute values with 2.75. For detecting the convergence, if the number of mismatches is less than 1% of the block size, then switching occurs.

5.2. Summary of results

Figures 5 and 6 shows the bit error rates (BER) of the proposed switching methods after the 5-*th* and 10-*th* iterations. At first, the switching turbo equalizer shows better BER than turbo equalizers employing $\mathbf{h}_1$ or $\mathbf{h}_2$ over all iterations. It is also observed that the methods based on soft information characteristics (SNR and average based methods) show better BER than others. The static method works well in a high SNR region, but not in the low SNR region. The BER of the convergence test method is close those of SNR or average based methods after the 10-*th* iteration, but is worse in earlier iterations.

From a VLSI implementation perspective, the SNR-based method is the most complex due to the need for the square and $\log$ functions. The convergence test needs extra storage. However, the other methods can be easily implemented using an accumulator, a counter, and a comparator with negligible overhead in comparison the turbo equalizer as a whole. Thus, the average method provides the best BER performance with low complexity.

6. REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. of IEEE Int. Conf. on Comm., Geneva*, May 1993, pp. 1064–1070.

[2] C. Douillard et al., "Iterative correction of intersymbol interference: Turbo equalization," *European Trans. on Telecommunication,* vol. 6, pp. 507–511, September-October 1995.

[3] A. Glavieux, C. Laot, and J. Labat, "Turbo equalization over a frequency selective channel," in *Proc. of Int. Symp. on Turbo Codes & Related topics,* September 1997, pp. 96–102.

[4] M. Tüchler, R. Koetter, and A. Singer, "Turbo-equalization: principles and new results," *IEEE Trans. on Comm.,* vol. 50, no. 5, pp. 754–767, May 2002.

[5] M. Tüchler, A. Singer, and R. Koetter, "Minimum mean squared error equalization using a-priori information," *IEEE Trans. on Signal Processing*, vol. 50, pp. 673–683, Mar. 2002.

[6] M. Tüchler, R. Koetter, and A. Singer, "Hybrid equalization strategies for iterative equalization and decoding," *Proc. of ISIT,* June 2001, pp. 267.

[7] A. J. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 260–264, Feb. 1998.

[8] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. on Comm.,* vol. 49, pp. 1727–1737, October 1999.