Synchronization of Turbo Codes Based on Online Statistics

Jian Sun and Matthew C. Valenti
Wireless Communications Research Laboratory
Lane Dept. of Comp. Sci. & Elect. Eng.
West Virginia University
Morgantown, WV 26506-6109
email: [jian,mvalenti]@csee.wvu.edu, phone: 304-293-0405, fax: 304-293-8602

***Abstract*— Turbo codes are sensitive to both (timing) synchronization errors and signal-to-noise ratio (SNR) mismatch. Since turbo codes are intended to be deployed in environments with very low SNR, conventional synchronization methods often fail. This paper introduces a solution for jointly estimating the SNR and achieving timing synchronization based on the statistics of the received signal. Simulation results show only a small loss in coding gain relative to perfect timing and SNR estimation while requiring only slightly more complexity and latency.**

I. Introduction

Turbo codes are capable of remarkable performance in very low signal-to-noise ratio (SNR) environments [1]. However, the full potential of turbo codes is only achieved if the channel statistics are known by the receiver. In an additive white Gaussian noise (AWGN) channel, the receiver must not only estimate the SNR of the channel, but also must synchronize with the bit epochs. While timing synchronization is an issue for any digital transmission system, it is especially important for turbo codes which operate at SNRs that are often too low for conventional synchronization techniques to work reliably. The result of a poorly synchronized signal is equivalent to additional AWGN noise [2], and for many practical situations the additional noise will cause a significant loss in coding gain.

Despite the importance of synchronization for turbo codes, this topic has been largely overlooked by the research community. While there currently exist a handful of publications on turbo code synchronization (see, for instance, [2], [3]), most studies of turbo codes assume perfect timing. Furthermore, papers that address the problem of timing synchronization tend to approach the problem separately from the problem of SNR estimation, and typically assume that the SNR is already known. However, these two problems are interrelated and thus a combined approach could be preferable. In this paper, we present a method for jointly estimating the SNR and the timing of the received signal. The strategy is an extension of the SNR estimation work of Summers and Wilson [4], which has been modified to account for imperfect timing.

At very low SNR, traditional symbol-by-symbol synchronization methods typically fail [5]. The high variance of the timing error prevents the front-end synchronization scheme from converging [3]. Thus, the approach we take performs synchronization on a frame-by-frame basis. Without a synchronization algorithm, the matched filter will (almost always) not be sampled at the proper sample instant. One approach to synchronization is to use the timing estimate to adjust the sample timing directly. However, this requires complex hardware and is not easily implemented. An alternative approach that we use is to oversample the output of the matched filter and store the samples in memory. Rather than using the timing estimate to adjust the sampling hardware, the estimate is used along with the stored samples to construct an estimate of the matched filter output at the optimal sample instant. This estimate is created by using an appropriate interpolation algorithm.

Mielczarek [2] introduced a soft-bit combining method for turbo code synchronization that employs two separate decoders and uses a weighted combination of the two decoder's soft outputs to generate a new likelihood value for each data bit. While this scheme provided performance within 0.2 dB of that with perfect timing, it did so at the cost of two decoders and sampling at four times the symbol rate and it assumed perfect SNR estimates. This complexity might not be affordable for many receivers. Furthermore, [2] only assumed a framesize of 256 information bits and a code rate of 1/2, which translated to a very weak turbo code that operated at high SNR where timing synchronization is less challenging. The goal of our study is to develop a synchronization algorithm with performance that is comparable to the one proposed by Mielczarek but which requires only a single turbo decoder and is able to operate with turbo codes with longer blocklength and lower code rate.

The remainder of this paper is organized as follows: Section II presents the system model, while Section III discusses our approach to combined synchronization and SNR estimation. Section IV gives simulation results, and Section V concludes the discussion.

II. System Model

To start, consider the traditional matched-filter receiver shown in Fig. 1, which is followed by a standard turbo decoder. Assume that BPSK modulation is used with (root) raised cosine-rolloff pulse shaping. Also assume that the receiver has perfect phase and frame synchronization information available and that the channel is quasi-static in the sense that it behaves as an AWGN channel for the duration of the frame and that

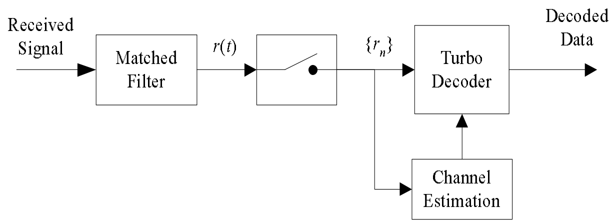

Fig. 1. Conventional receiver structure including a channel estimator and turbo decoder

the timing offset is constant for the entire frame (although the channel SNR and timing offset may vary from frame to frame).

The receiver's matched filter has the transfer function of a (root) raised cosine rolloff pulse. The output of the matched filter is $r(t)$, and the samples taken by the sampler are $r_n = r(nT + \tau)$, where τ is the timing offset and T is the symbol duration. With perfect timing, i.e. $\tau = 0$, the output of the matched filter has no intersymbol interference (ISI), and the SNR of the samples is exactly E_s/N_0, where E_s is the symbol energy and $N_0/2$ is the two-sided power spectral density of the AWGN (if the code rate is R, then the energy per information bit E_b is E_s/R). However, when perfect timing is not available, the performance may degrade significantly. For raised cosine rolloff pulse shaping, the performance degradation is not only due to the loss in received signal power, but also due to the presence of rather severe ISI. Thus the effective SNR in the presence of imperfect timing should account for both the loss of signal energy and the ISI.

The mean squared error from non-perfect timing is [6]

$$I(\tau) = \sum_{k=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} m_{k-j}x(\tau - kT)x(\tau - jT) - 2\sum_{k=-\infty}^{\infty} m_k x(\tau - kT) + m_0, \quad (1)$$

where $x(t)$ is the raised cosine rolloff pulse shape function in the time domain [7], and m_k is the autocorrelation of the coded sequence $\{a_k\}$,

$$m_k = \mathbf{E}\left[a_i a_{i+k}\right].$$

Assuming that the a_k's are independent and zero-mean, the autocorrelation of the coded sequence is

$$m_k = \begin{cases} E_s & \text{if } k = 0 \\ 0 & \text{if } k \neq 0. \end{cases} \quad (2)$$

Therefore, the mean squared error is

$$I(\tau) = E_s\left[\sum_{k=-\infty}^{\infty} x^2(\tau - kT) - 2x(\tau) + 1\right]. \quad (3)$$

This function has a parabolic shape with $I(0) = 0$. Thus, the function can be approximated with power series expansion near the origin ($\tau = 0$), i.e.

$$I(\tau) = AE_s\tau^2 + (\text{higher-order terms}) \quad (4)$$

where A is the second order Taylor expansion coefficient

$$A = \frac{1}{2E_s}\frac{d^2 I(\tau)}{d\tau^2}\bigg|_{\tau=0}. \quad (5)$$

When the roll-off factor is 0.5, the value of A is 1.3.

Mielczarek [2] models the ISI as an additive Gaussian noise independent of the channel noise. More specifically, the effective SNR β can be expressed as

$$\beta = \frac{x(\tau)E_s}{2\left(E_s A\tau^2 + N_o/2\right)} = \frac{x(\tau)E_s/N_o}{2A\tau^2 E_s/N_o + 1}. \quad (6)$$

III. Joint Timing-Offset/Channel-SNR Estimation Algorithm

Because (6) is a function of τ, it could be used to obtain an estimate of the timing-offset $\hat{\tau}$. Thus, a reasonable method for computing $\hat{\tau}$ would be to first estimate the effective SNR β and then invert (6) to obtain the timing-offset estimate. However, a direct implementation of this approach also requires knowledge of E_s/N_o, which we assume is not known. Thus, a more indirect approach for simultaneously estimating τ and E_s/N_o is desired. Our approach to simultaneously estimating τ and E_s/N_o is to use *two* matched filter samples per symbol interval (sampled $0.5T$ apart). Because now there are essentially two equations and two unknowns, it is possible to determine which set of τ and E_s/N_o estimates produces an effective SNR function β which best fits the two samples over the entire frame.

Successful implementation of this strategy requires fairly accurate estimates of the effective SNRs for each of the two sample positions averaged over the entire frame. To compute the effective SNR estimate, we use the approach proposed by Summers and Wilson [4] which requires the computation of an online statistic using the sample means of r_n^2 and $|r_n|$, i.e.

$$s = \frac{E\left[r_n^2\right]}{E\left[|r_n|\right]} = \frac{1 + 2\beta}{\sqrt{\frac{2}{\pi}e^{-\beta} + \sqrt{2\beta}\,\mathrm{erf}\left(\sqrt{\beta}\right)}} = f(\beta) \quad (7)$$

We are interested in the inverse function $\beta = f^{-1}(s)$, from which we can get the estimated effective SNR for each of the two sample positions. This inverse function is implemented with a look-up table (LUT). Fig. 2 shows the analytical value of β for $E_s/N_0 = 0$ dB as a function of timing offset. In addition, the curve also shows the estimated value of β found by inverting (7) when simulating a frame of length 4590 (100 simulation runs were averaged to generate this curve). Two points should be noted. First, the simulated results agree closely with the analytical value for all but the most extreme timing offsets. Second, the loss in effective SNR can be in excess of a decibel for timing offsets greater than $0.18T$. For turbo coded systems operating at low SNR, this loss will render the decoder useless.

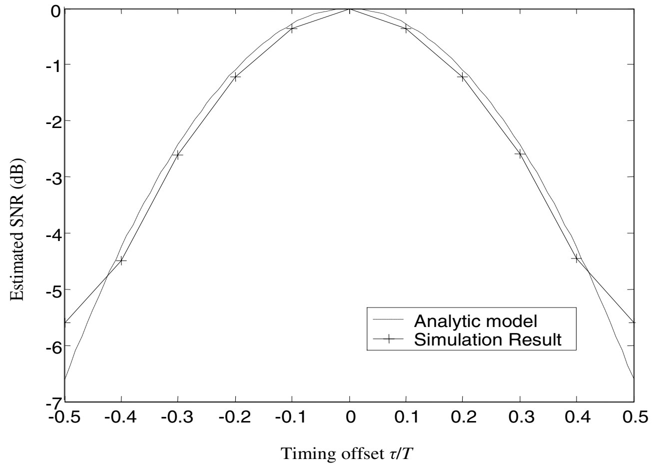

Fig. 2. Effective SNR in the presence of improper timing, both using the analytic model and simulatd results (100 trials with a code frame size 4590). The actual channel SNR is 0 dB.

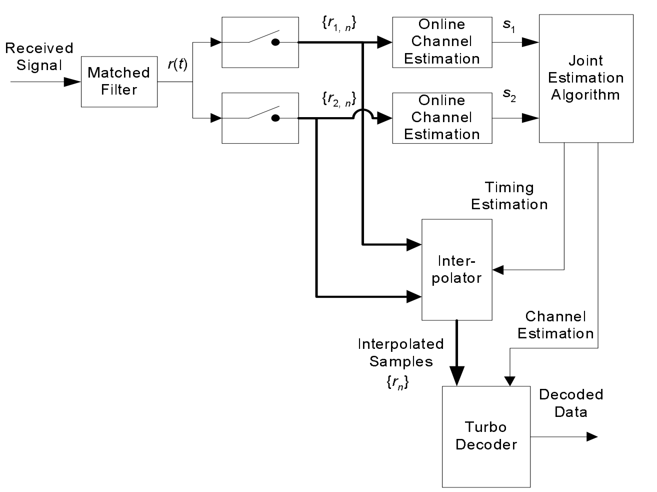

Fig. 3. Modified turbo coded system with multiple sampler, estimation, and interpolator blocks

The process of simultaneously estimating the timing offset and the channel SNR is summarized by the system diagram shown in Fig. 3. The matched filter output is sampled twice, the first at time $t = nT + \tau - 0.5T$ and the second at time $t = nT + \tau$. i.e.

$$\begin{aligned} r_{1,n} &= r(nT + \tau - 0.5T) \\ r_{2,n} &= r(nT + \tau). \end{aligned}$$

Without loss of generality and assuming perfect frame synchronization, it is assumed that $0 \leq \tau < 0.5T$ such that the first sample $r_{1,n}$ occurs during the interval $[(n - 0.5)T, nT)$ and the second sample $r_{2,n}$ occurs during $[nT, (n + 0.5)T)$.

For each of these two sample positions, a sample expectation is taken of $|r_{i,n}|$ and $r_{i,n}^2$, $(i = 1, 2)$, over the entire frame inside the "online channel estimation" blocks, which are used to compute the corresponding online statistic s_1 and s_2. These values are fed into the "joint estimation algorithm" block which uses $f^{-1}(s)$ to estimate the effective SNR for each sample position. When E_s/N_o is known, (6) can be inverted to produce an estimate of the timing offset. Due to the finite length of the received frame, the estimated means of $|r_{i,n}|$ and $r_{i,n}^2$, $(i = 1, 2)$ may differ from the expected values. This error has a strong influence on the overall performance of the system, as it will cause the timing estimate to differ from its actual value. Fortunately, Fig. 2 begins to flatten near its peak at $\tau = 0$, indicating that residual timing offset errors of $|\tau| < 0.1T$ will not significantly impede performance.

When the channel SNR E_s/N_o is not known, then the two estimated effective SNR's β_1 and β_2 can be used to jointly estimate the timing offset and channel SNR. This joint estimation is possible because the β curve is unique for each value of the actual channel SNR. In other words, if the β function were drawn for multiple values of E_s/N_o, then these curves would not touch each other. Therefore, two estimated effective SNR values are sufficient to determine both which curve (i.e. estimate the SNR) and determine the timing offset, provided that the time difference between the two estimated effective SNR values is known (i.e. $0.5T$).

Because there is a one-to-one mapping between online statistics $\{s_1, s_2\}$ and estimated effective channel SNRs $\{\beta_1, \beta_2\}$, the joint estimation algorithm could work directly with the online statistics. Thus, our simplified estimation algorithm uses s_1 and s_2 to determine the following set of functions:

$$\begin{aligned} f &= \operatorname{sign}(s_1 - s_2) \\ b &= \min(s_1, s_2) \\ d &= |s_1 - s_2|, \end{aligned} \tag{8}$$

where f denotes the direction of bias, and the combination of b and d defines the relationship of the values s_1 and s_2. The algorithm also requires two thresholds T_1 and T_2, which are selected empirically and used to classify the initial synchronization state.

If the timing-offset of either sample is close to zero, then the other sample will have an offset close to $\tau = \pm 0.5T$. When this situation occurs, the difference d between the two online statistics will be quite large. Thus, if $d > T_1$ it is assumed that either τ or $(\tau - 0.5T)$ are close to zero, i.e. one of the two samples is almost perfectly synchronized. The selection between τ and $(\tau - 0.5T)$ would be determined by f, i.e. the sample with the higher online statistic is assumed to be close to synchronization.

Due to the symmetry of Fig. 2, the two online statistics will have roughly equal value when the optimal sample instant is halfway between the two sample instances, i.e. when $\tau = 0.25T$. Thus, when $d < T_2$ the timing offset is set to $\hat{\tau} = 0.25T$. When $d \not\approx 0$ or $\tau \not\approx 0.25T$, then the timing-offset estimate is set to:

$$\hat{\tau} = \left(\frac{1}{4} - \frac{fd}{4T_1} \right) T \tag{9}$$

Because the slope of the curve generated by (6) varies with the channel SNR, the thresholds T_1 and T_2 should be functions of b. Hence two look-up tables are established for T_1 and T_2 with entries relative to b.

The timing estimation value $\hat{\tau}$ controls the interpolator which reconstructs the samples with perfect timing according to

$$r_k = 2\hat{\tau} r_{1,k} + (1 - 2\hat{\tau})\, r_{2,k} \quad (10)$$

Note that this is a simple linear interpolation rule. Although our goal was to achieve a simple synchronization algorithm (albeit a somewhat heuristic one), better performance could be attained by using a more sophisticated interpolator which takes into account the pulse shaping. Strictly speaking, the bandwidth when using raised-cosine pulse shaping is greater than $1/T$, so the signal is sampled at less than the Nyquist rate when just two samples are used per symbol. Thus, a more sophisticated interpolator would benefit from a higher sampling rate to avoid aliasing. However, we found little additional gain by using a more precise interpolator. The negligible gain from using more sophisticated interpolation can be attributed to the low SNR. For example, when E_b/N_0 is 0 dB and $R = 1/3$, the SNR per symbol E_s/N_0 is -4.77 dB at which point the waveform $r(t)$ becomes dominated by noise. Therefore, the sample values are not reliable enough to reconstruct the interpolation values.

The joint estimatation approach can be easily extended to the situation that the matched filter ouput is oversampled N times each symbol. If $N > 2$, then the 2 samples with highest effective SNR's are selected because of the fact that the sample closer to perfect timing has higher effective SNR. The time difference between the samples is then T/N.

IV. Simulation Results

A set of computer simulations were performed to investigate the performance of the proposed joint estimation algorithm. The turbo code and its interleaver were designed according to the cdma2000 specification [8], with the rate set to $R = 1/3$ and interleaver size set to 1530.

Fig. 4 shows the BER performance of the turbo code with several fixed timing offsets. This figure gives a general idea about the effect of timing shift. The performance gap the perfect timing curve and the other curves with nonzero timing offset grows exponentially (in dB) with τ. At a BER of 10^{-5}, a timing offset of only $\tau = 0.2T$ would create a loss in coding gain of more than 1 dB.

The BER performance with the modified system structure described in Fig. 3 is shown in Fig. 5. Four curves are shown corresponding to (from most to least energy efficient): (1) Perfect timing ($\tau = 0$) and perfect SNR estimation (because timing is perfect, only one sample per symbol is needed), (2) Imperfect, but known timing (τ is uniformly distributed in $[0, 0.5T]$) and perfect SNR estimation, (3) Imperfect and unknown timing (which is estimated) but known SNR, and (4) Both timing and SNR are unknown and estimated.

When the timing is imperfect, but both the timing offset and channel SNR are known to the receiver, the performance is within 0.2 dB from the perfect timing case when the BER is 10^{-5}. Since all estimation blocks are disabled and perfect knowledge of timing and SNR are used, the loss comes completely from the simple linear interpolation method.

When the channel SNR is known but timing must be estimated, the loss relative to perfect timing increases to approximately 0.7 dB at a BER of 10^{-5}. While 0.2 dB loss is attributed to the linear interpolation, the rest comes from the timing estimation. When the channel SNR is known, the timing estimation algorithm is much simpler than that described at the end of the previous section. In particular, the timing offset is determined by the differences of the estimated effective SNR values and the actual channel SNR. Let $\hat{\beta}_1$ and $\hat{\beta}_2$ be the estimated effective SNR values from sample sets $\{r_{1,n}\}$ and $\{r_{2,n}\}$ respectively and β be the actual SNR, all

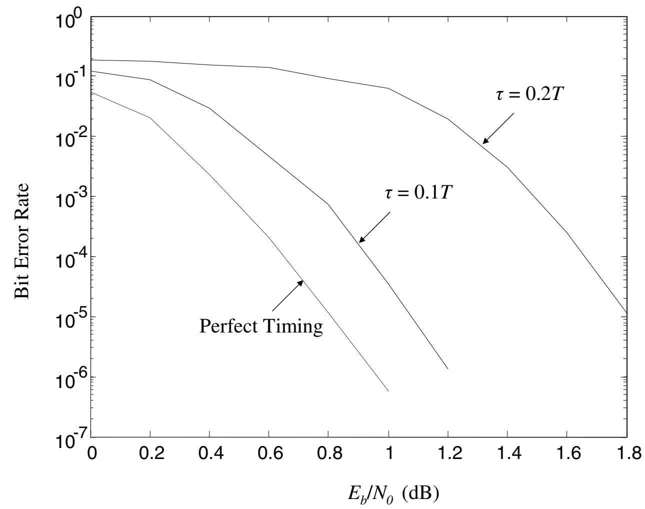

Fig. 4. Simulation results showing cdma2000 turbo code performance for various fixed timing offsets (interleaver size = 1530, rate $R = 1/3$, 10 decoder iterations, and BPSK modulation with RC-rolloff pulse shaping).

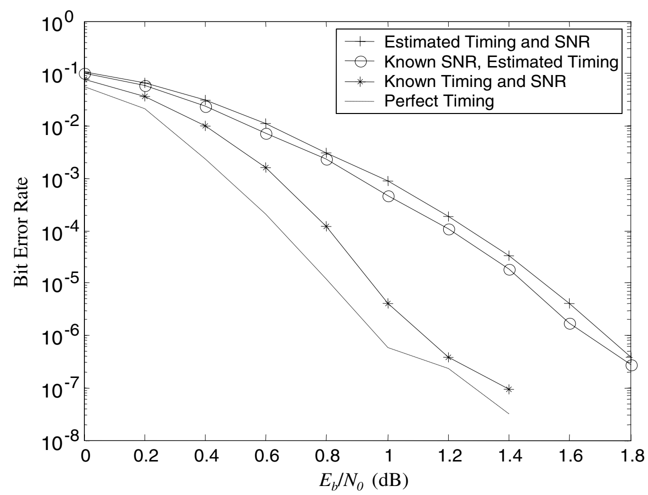

Fig. 5. Bit error rate performance of the cdma2000 turbo code with different types of frame-by-frame synchronization (2 samples per symbol, interleaver size = 1530, rate $R = 1/3$, 10 decoder iterations, and BPSK modulation with RC-rolloff pulse shaping).

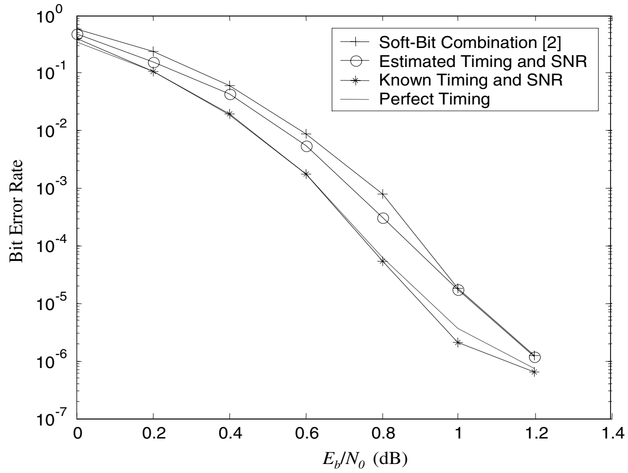

Fig. 6. Bit error rate performance of the cdma2000 turbo code with different types of frame-by-frame synchronization (4 samples per symbol, interleaver size = 1530, rate $R = 1/3$, 10 decoder iterations, and BPSK modulation with RC-rolloff pulse shaping).

in dB, then the timing offset is determined using

$$\frac{0.5-\hat{\tau}}{\hat{\tau}} = \frac{\beta-\hat{\beta}_1}{\beta-\hat{\beta}_2} \qquad (11)$$

While lack of channel SNR information causes an additional performance loss, this loss is actually rather small. This is consistent with the results of [4], which indicates a small loss when only estimating the SNR in the presence of perfect timing. Thus, most of the overall system loss is due to the timing estimation. With larger frame sizes, both the timing and channel estimation will become more reliable, and the coding gain loss should decrease.

Fig. 6 presents the BER performance of the joint esimation algorithm with 4 samples per symbol. Four curves are shown corresponding to (from most to least energy efficient): (1) Perfect timing ($\tau = 0$) and perfect SNR estimation, (2) Imperfect, but known timing (τ is uniformly distributed in $[0, 0.5T]$) and perfect SNR estimation, (3) Both timing and SNR are unknown and estimated, and (4) Post-decoding synchronization using the soft-bit combining algorithm developed by Mielczarek [2].

When timing is imperfect, but both the timing offset and channel SNR are known to the receiver, the loss of coding gain is negligible. This means, with 4 samples per symbol, there is approximately no loss from the simple linear interpolation method. When no knowledge about timing offset and channel SNR is available to the receiver, the coding gain loss at a BER of 10^{-5} is about 0.1 dB. This is a significant improvement comparing to the coding gain loss with only 2 samples per symbol. The reason is that the samples selected have higher effective SNR and the timing estimation has less variance. At very low SNR, i.e. when E_s/N_0 is within 1 dB, the joint estimation algorithm yields better performance than the soft-bit combining method, and at higher SNR, the two curves merge together. Note that joint estimation approach only uses one turbo decoder while soft-bit combining requires two, the overall system complexity of the former is greatly less than the latter.

V. Conclusions

Analytical and simulation results indicate that imperfect timing causes a loss in effective SNR which results in a severe BER performance degradation for timing shifts greater than about 10% of the symbol period. This performance loss can be recovered by a proper estimation algorithm. However, the situation is complicated by the fact that the channel SNR over which turbo codes operate are both very small and not known to the receiver. Our approach involves sampling the signal multiple times per symbol period and computing an online statistic for each of the sample instances over the entire frame. These online statistics are then used to simultaneously estimate the channel SNR and timing offset. A simple linear interpolation algorithm is then used to reconstruct the matched filter samples at the estimated timing instants.

The proposed algorithm recovered much of the loss due to poor synchronization, and did so with negligible added complexity and latency (compared to that of the turbo decoding algorithm itself). The simulated coding gain loss is about 0.1 dB with 4 samples per symbol. Modifications to the algorithm can further close this gap. Suggested improvements include:

1) A less heuristic timing estimation algorithm which mathematically fits the online statistics $\{s_1, s_2\}$ with the set of $\{E_s/N_o, \tau\}$ indicated by (6). Linear MLSE algorithm, using curve fitting, has been applied, but the performance was actually worse. The proposed heuristic approach may be beneficial, because of the nonlinearity of (6),
2) Exploitation of the turbo principle, i.e. pass information from the turbo decoder back to the timing/channel estimation algorithm and use this information to reestimate the timing and SNR after each iteration. Negligible coding gain loss has been achieved with 4 samples per symbol and only one global iteration.

References

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

[2] B. Mielczarek, *Synchronization in Turbo Coded systems*. PhD thesis, Chalmers University of Technology, Sweden, April 2000.

[3] L. Lu and S. G. Wilson, "Synchronization of turbo coded modulation systems at low SNR," in *Proc., IEEE Int. Conf. on Commun.*, pp. 428–432, 1998.

[4] T. A. Summers and S. G. Wilson, "SNR mismatch and online estimation in turbo decoding," *IEEE Trans. Comm.*, vol. 46, pp. 421–423, April 1998.

[5] H. Meyr and G. Ascheid, *Synchronization in digital communications*. Wiley series in telecommunications, John Wiley and Sons, Inc., 1990.

[6] L. E. Franks, "Further results on Nyquist's problem in pulse transmission," *IEEE Trans. Comm. Tech.*, vol. 16, pp. 337–340, April 1968.

[7] J. G. Proakis, *Digital communications*. McGraw-Hill, Inc., 3 ed., 1995.

[8] *Physical Layer Standard for cdma2000 Spread Spectrum Systems, Release B*. 3GPP2, April 2002. Available at http://www.3gpp2.org.