Iterative APPA Symbol Timing Recovery for Turbo-coded Systems

Li Zhang and Alister Burr

***Abstract*— This paper examines the problem of symbol timing recovery and decoding in turbo-coded systems. The large coding gain of Turbo-codes enables reliable communications at very low signal-to-noise ratio (SNR). However, the application of turbo coding exacerbates the problem of timing recovery due to its very low operating SNRs. Therefore we propose a new concept: '*a-priori* probability aided' symbol timing recovery, which is suited to coding systems employing iterative soft-in/soft-out (SISO) decoding such as Turbo Codes, where the log-likelihood ratio obtained from a Maximum A Posteriori (MAP) decoder (as used in turbo-decoding) is used to aid estimation in an iterative synchroniser/decoder. This method jointly solves the timing recovery and decoding problem and the synchroniser has effectively no acquisition period. We illustrate this in detail in a turbo-coded BPSK system on AWGN channel. We analyse the performance of the estimator in terms of the mean and the variance of the timing estimate, and demonstrate a substantial performance advantage over conventional non-data-aided (NDA) method. It is also shown that this method can approach the performance of Data-Aided (DA) method without using additional pilot symbols.**

***Index Terms*—** *a priori information, timing offset, turbo codes, S-curve, AWGN channel*

I. INTORDUCTION

TURBO-CODES [1] are by now well known as very powerful error control codes, which are capable of closely approaching the Shannon bound on channel capacity. However, the impressive performance of turbo codes always implicitly assumes perfect synchronisation, which is unrealistic. In practical communication systems performance may often be limited not so much by errors due to noise as by the problem of synchronisation [2]. Two main synchronisation parameters in most communication systems are carrier phase and symbol timing. Precise knowledge of the received carrier phase and accurate symbol timing are necessary for reliable detection in communication systems. In [3], we have addressed the carrier phase recovery problem assuming perfectly known symbol timing in turbo coded systems. In this paper, we will investigate the problem of symbol timing recovery assuming perfectly known carrier phase.

Timing recovery is one of the most critical synchronisation functions that are performed at the receiver of a digital communication system. Sampling at optimum instants to ensure a minimum intersymbol interference (ISI) is crucial to achieving good overall performance. The application of coding, and especially turbo-coding, tends to exacerbate the symbol timing recovery problem, precisely because of its low operating SNR. Moreover, the turbo-coded system is rather sensitive to a timing offset, and this is in effect because the timing offset reduces the signal power and simultaneously increases the variance of the ISI at the output of the matched filter and thus reduces the effective SNR significantly. Conventional non-data-aided (NDA) timing recovery techniques are dependent on SNR: coded systems operate at a lower bit energy to noise density ratio than uncoded, by virtue of their coding gain, and further at a lower still SNR, because of their redundancy. Data-aided (DA) or decision-directed (DD) timing recovery, on the other hand, requires either long preambles, which increase redundancy, or access to decoding decisions, which are generally not available until synchronisation has been performed.

In the literature, some effort has been devoted to the development of efficient symbol timing recovery methods suited to turbo-coded systems. Lu *et al.* [4] present a timing offset tracking algorithm in turbo-coded modulation systems, but they use the conventional early-late gate algorithm [5] without any help from the turbo decoder. Mielczarek *et al.* carried out a programme of research on this issue. Their most recent work [6] proposes a sort of *soft bit combination* timing recovery approach taking account of the influence of timing offset on turbo decoding. In this approach, a coarse ML NDA detector selects two sets of samples which lie closest to the optimum sampling point from four samples per symbol, and sends to the decoding process. They are decoded for enough iterations and combined to create final soft bit values used in thresholding device. Another work of Mielczarek [7] obtains the timing error according to a look-up table, which links the timing errors to the difference of the average squared Log-Likelihood Ratio (LLR)(output from turbo decoding) between two samples taken equally separated from the optimum sampling point. Both approaches employ two turbo decoders and result in a high complexity.

In this paper, we describe an alternative concept, which we will call *a priori* probability aided (APPA) timing recovery, which fits particularly well with the iterative turbo-decoder, although it would also be applicable with other types of code. It can be understood as a generalisation of DA and NDA synchronisation, since it reduces to the former when perfect data knowledge is available, and to the latter when there is no *a priori* knowledge of the data. In place of hard data decisions,

This work is funded by QinetiQ under the MoD applied research programme. Li Zhang is with the School of Electronics and Electrical Engineering, University of Leeds - Email: l.x.zhang@leeds.ac.uk. Alister Burr is with the Department of Electronics, University of York - Email: alister@ohm.york.ac.uk. This paper was presented in part at the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications PIMRC'2002, Lisbon, Portugal, Sept. 2002.

ISSN 1980–6604/$20.00

as in the DD case, it uses the *a priori* probability (APP) information obtained from the decoder, which carries the probability information of the transmitted data from the output of a SISO decoder, such as the MAP decoder used for turbo-codes. Note that the term '*a priori*' is used because its well-established use in turbo decoding and because the information is *a priori* as far as the synchronisation is concerned; in fact the information is *a priori* only in respect of the next iteration of the decoder - it is *a posteriori* in relation to the original data. The timing estimation and the turbo decoding are performed jointly once per decoding iteration. The output timing estimate and the extrinsic information are then used in the next iteration. Applied iteratively, this technique allows successive refinement of the symbol timing estimate, until the joint decoder/synchroniser converges on the (hopefully) correct data and timing estimate. In this way, this iterative timing recovery technique exploits the power of turbo codes to enable reliable operation at very low SNR. There is no need for transmission of additional pilot symbols, and hence it also saves bandwidth. A similar proposal is made in [8], where the timing recovery and equalisation are implemented iteratively while interacting with the decoder. But this work is for the low-density parity-check (LDPC) codes assuming that the timing error may be modelled by a Markov chain.

The proposed symbol timing estimation method is based on the Maximum Likelihood algorithm [9] with low complexity. The most likely estimate of the timing error maximises the log-likelihood function, and corresponds to the zero crossing of the negative slope of the S-curve [10], [11] (which is defined as the derivative of the log-likelihood function). Accordingly, we obtain the maximum estimate directly from the S-curve without an acquisition procedure. In addition, we obtain this zero crossing by linear interpolation based on four samples at -T/2, -T/6, T/6, T/2 on the S-curve, without producing the whole S-curve, and hence significantly reduce the computational complexity.

We address the symbol timing recovery in turbo-coded BPSK system on the AWGN channel in this paper and initially assume the perfectly known carrier phase (set as 0). In the next section, we derive the log-likelihood function for the NDA, APPA and DA methods, and then describe how to obtain the maximum likelihood estimate with reduced complexity. The third section describes the receiver structure. While in the fourth section we give simulation results and compare the APPA method with the other two conventional methods. Finally, the last section gives our conclusions.

II. APPA Symbol Timing Estimator

A. System Model

In this paper, we consider a turbo-coded BPSK system. The turbo encoder is built using a parallel concatenation of two identical recursive systematic convolutional (RSC) codes with generators $[G_1, G_2]$ linked together by an interleaver with size N, where G_1 and G_2 are the polynomials of the feedback and output connectivities of the RSC encoders. Both RSC encoders use the same information data bits but according to a different sequence due to the presence of the interleaver. The parity bits out of the two encoders are properly punctured to achieve the desired coding rate. The coded streams are modulated to BPSK symbols and then pulse shaped using a band-limited signal pulse $g(t)$ and the resulting baseband signal is transmitted over an AWGN channel.

The input signal to the channel can be expressed in complex baseband form as

$$x(t) = \sum_k d_k g(t - kT) \tag{1}$$

where $\{d_k\}$ is a sequence of random binary data symbols which take on the values ± 1 with equal probability. $1/T$ is the symbol rate and $g(t)$ is the Nyquist signalling pulse shape.

The received signal is then sampled and filtered by a matched filter. To sample at the optimum instants, the receiver must know not only the frequency $1/T$ at which the outputs of the matched filters or correlators are sampled, but also where to take the samples within each symbol interval, which is called *timing offset* and denoted as τ. In an analogue receiver the solution of this problem is to control the sampling instant $kT + \tau$ of the received signal [12], [13]. In truly digital timing recovery, sampling is done using a fixed clock at $t = kT_s$ possibly incommensurate with the symbol rate $1/T$. The shifted samples must be obtained solely from those asynchronous samples taken at rate $1/T_s$ rather than shifting a physical clock. Its distinct feature is that the clock is not adjusted in any way; proper timing must be established from signal samples themselves. If the Nyquist sampling criterion is met, the samples will contain sufficient information to allow reconstruction of the underlying signal at the optimum strobe point through interpolation. In this paper, we consider a fully digital implementation. For simplicity, we assume the sampler at receiver operates at the same rate $1/T$ as at the transmitter. Then the remained problem is to determine the correct timing offset within a T-second interval $[-T/2, T/2]$ [10].

Assuming perfect carrier recovery, the received complex baseband signal in the AWGN channel can be expressed as

$$r(t) = x(t + \tau) + n(t) \tag{2}$$

where $n(t)$ is white Gaussian noise with power σ^2, and τ is the unknown but deterministic timing offset, which implies that Maximum Likelihood Estimation (MLE) is the most suitable estimation method [9]. In addition, we assume the timing offset remains constant through the coding block. Since in the majority of the existing applications the block length is much shorter than the timing offset fluctuation period, this assumption does not introduce significant penalty.

The key idea in developing an efficient symbol timing recovery scheme in a turbo-coded system is to take advantage of the iterative structure of the turbo decoding by interacting with the decoder [14]. In the following, we will show how to implement timing estimation with help from the decoder.

B. Derivation of the Log-Likelihood Function

The approaches discussed here are based on the maximum-likelihood strategy for estimation of the parameter τ in

(2) based on observations of the received signal. The log-likelihood function (LLF) is

$$\Lambda(\tau) = \frac{1}{\sigma^2}\int_0^{NT} r(t)x(t+\tau)dt \tag{3}$$

where N is the observation interval, which is the same as the number of data bits in one block of turbo codes. Note that the extrinsic information obtained from the Turbo decoder is only available for the information data bits, hence we only consider the data bits in the calculation of the LLF. Since the turbo decoder processes data blocks, taking a block of data for every synchronisation observation will not introduce further delay.

In term of the received signal, we can write the LLF of DA estimation as

$$\begin{aligned}\Lambda_{da}(\hat{d},\tau) &= \frac{1}{\sigma^2}\int_0^{NT} r(t)\sum_{k=0}^{N-1}\hat{d}_k g(t-kT+\tau)dt \\ &= \sum_{k=0}^{N-1}\hat{d}_k q_k(\tau)\end{aligned} \tag{4}$$

where $\hat{d}_k$ is the known data or the data decision made in the receiver and $q_k(\tau)$ is given by

$$q_k(\tau) = \frac{1}{\sigma^2}\int_0^{NT} r(t)g(t-kT+\tau)dt. \tag{5}$$

Considering the transmitted data as a *nuisance* parameter, the likelihood function in NDA timing estimation can be obtained by averaging the likelihood function in the DA estimation over the PDF of the information symbols with distribution $p_r(d_k = \pm 1) = 1/2$ for the baseband binary signal. Hence the LLF of the NDA method is

$$\begin{aligned}\Lambda_{nda}(\tau) &= \ln\prod_{k=0}^{N-1}\left[\frac{1}{2}\exp\left(q_k(\tau)\right)+\frac{1}{2}\exp\left(-q_k(\tau)\right)\right] \\ &= \sum_{k=0}^{N-1}\ln\cosh\left(q_k(\tau)\right).\end{aligned} \tag{6}$$

In this paper, the crucial point is how to take advantage of the structure of the turbo decoder in a turbo-coded system. Notice the most significant feature of turbo-codes is the iterative decoding, which uses two SISO component decoders for the two component encoders. The soft output (extrinsic information) from the first decoder will be used as the *a priori* information in the second decoder, while the extrinsic information generated from the second decoder will be used as the *a priori* information in the first decoder in the next iteration, and so on. In this way, each decoder takes advantage of the extrinsic information produced by the other decoder in the previous step. The *a priori* information is the estimation of the transmitted data made by the component turbo decoder, meaning how likely the data is '1' or '-1', expressed mathematically as [15]

$$L(k) = \log\left(\frac{p(d_k=1)}{p(d_k=-1)}\right). \tag{7}$$

The more positive this value, the more likely the data is to be '1', on the other hand, the more negative this value, the more likely it is to be '-1'. The absolute magnitude of $L(k)$ is the measure of the reliability of the estimation: the larger, the higher the probability of the data bit being 1, when $L(k)$ is positive; or -1, when it is negative. However, when $|L(k)|$ reduces to 0, that means the decoder is uncertain what the data is (i.e. $p(d_k = \pm 1) = 1/2$). This can be directly used for the distribution of the data and used in the synchroniser. In binary transmission, $p(d_k = 1) + p(d_k = -1) = 1$, and hence we can compute the symbol probabilities [15] from $L(k)$ by

$$\begin{aligned}p(d_k=1) &= \frac{\exp(L(k))}{1+\exp(L(k))} \\ p(d_k=-1) &= \frac{1}{1+\exp(L(k))}.\end{aligned} \tag{8}$$

Averaging the likelihood function in the DA estimator (the LLF is given in (4)) over all possible transmitted symbols: $S_0 = 1$ and $S_1 = -1$ in BPSK, using the probability information given in (8), we get the LLF of the timing estimator aided by the *a priori* information

$$\begin{aligned}\Lambda_{appa}(\tau) = \sum_{k=0}^{N-1}\log[&p(d_k=S_0)\exp\left(q_k(\tau)\right)+ \\ &p(d_k=S_1)\exp\left(-q_k(\tau)\right)].\end{aligned} \tag{9}$$

As a practical implementation consideration, we restrict the integration in $q_k(\tau)$ to $[-aT, aT]$ [16] and denote it as

$$f_k(\tau) = \frac{1}{\sigma^2}\int_{-aT}^{aT} r(t+kT)g(t+\tau)dt \tag{10}$$

where a is a parameter to determine the range of the integral: the larger the more accurate (except at the ends). For most purposes $a = 2$ or 3 will probably adequate. In our study we take a value of $a = 4$. Then the LLF becomes

$$\begin{aligned}\Lambda_{appa}(\tau) = \sum_{k=0}^{N-1}\log[&p(d_k=1)\exp(f_k(\tau))+ \\ &p(d_k=-1)\exp(-f_k(\tau))].\end{aligned} \tag{11}$$

Inserting (8) into (11), after some algebra, we get

$$\Lambda^l_{appa}(\tau) = \sum_{k=0}^{N-1}\log\left[\cosh\left(f_k(\tau)+\frac{L^l(k)}{2}\right)\operatorname{sech}\frac{L^l(k)}{2}\right]. \tag{12}$$

We notice that through this modified LLF, the *a priori* information $L(k)$ is embedded into the maximum likelihood timing estimation. We call it *a priori* probability aided (APPA) timing estimation. Note that the LLF for the APPA timing estimator is now a function of the extrinsic information calculated iteratively in the turbo decoding. We use l to indicate the number of decoding iterations, showing that the LLF of APPA estimation is updated iteratively along with the turbo decoding. Consequently, it will yield an updated timing estimate for each decoding iteration. The new timing estimate is used to correct the timing offset of the data sequence fed to the decoder, which will generate new extrinsic information. Based on this new extrinsic information, timing estimation is repeated, generating new timing estimates that are more accurate than the originals. Repeating timing estimation and decoding jointly, the timing estimation is performed multiple

times while interacting with the decoder. The hope is that each set of extrinsic information will improve the timing estimates, which will in turn improve the next set of data samples fed into the next decoding iteration, so that eventually the algorithm will converge to the optimal solution of the joint symbol timing estimation and turbo decoding.

As already mentioned above, the extrinsic information is the estimate of the transmitted data bit made by the previous decoding attempt. The absolute value of the extrinsic information $|L(k)|$ represents a sort of reliability metric, the larger it is the more reliable the estimate is, and the more the two probabilities are unbalanced. When it is large enough, one probability will be 1, the other 0, that is the decoder is very certain about the decision, and the APPA estimator resembles the case of the DA method. On the other hand, when the extrinsic information is equal to 0, the decoder makes an estimate with $p(d_k = 1) = p(d_k = -1) = 1/2$, which is exactly the same as the NDA case. However, this case will happen only at the first iteration, when no extrinsic information is available. Clearly, the APPA method lies between the DA and the NDA methods in terms of how much information about the transmitted data is available: with large $|L(k)|$, the APPA method approaches the DA method; while with small $|L(k)|$, it tends to NDA. As long as iterative decoding proceeds, the reliability of decoding improves, and the absolute values $|L(k)|$ increase. The iteratively improved *extrinsic information* will improve the proposed APPA method iteration by iteration to match the DA method. This will be verified by the simulation results in the following.

C. Maximum Likelihood Timing Estimate

The proposed timing recovery technique is based on choosing the value of $\hat{\tau}$ to maximise the LLF, given in (12) for the APPA method. Differentiating (12) with respect to τ, we obtain the necessary condition for maximising the LLF

$$\begin{aligned} & {\Lambda^l_{appa}}'(\tau) \\ = & \sum_{k=0}^{N-1} \frac{1}{\sigma^2} \int_{-aT}^{aT} r(t+kT) g'(t+\tau) dt \cdot \\ & \tanh\left(\frac{L^l(k)}{2} + \frac{1}{\sigma^2} \int_{-aT}^{aT} r(t+kT) g(t+\tau) dt \right) \\ = & \sum_{k=0}^{N-1} f'_k(\tau) \tanh\left(\frac{L^l(k)}{2} + f_k(\tau) \right) = 0 \end{aligned} \tag{13}$$

where

$$f'_k(\tau) = \frac{1}{\sigma^2} \int_{-aT}^{aT} r(t+kT) g'(t+\tau) dt. \tag{14}$$

$f'_k(\tau)$ can be interpreted as the output from filter with impulse response $g'(t+\tau)$, which is the derivative of $g(t+\tau)$ with respect to τ.

However, observing the highly complicated function (13), it involves too much computation to compute the exact estimate from (13) directly. To resolve this problem, we turn to the use of S-curve.

D. S-curves

The S-curve [10] is defined as the derivative of the LLF (12) with respect to τ, and hence the S-curve for the APPA method is given by (13). It is not difficult to deduce that the zero crossing of the negative slope of the S-curve maximises the LLF and it is the desired timing estimate. The S-curve can be regarded as a *discriminator characteristic* for measuring the parameter τ [10]. The estimation accuracy is enhanced by increasing the slope at the zero crossing.

Fig. 1(a) shows the S-curves obtained by simulation using the DA, NDA and APPA timing estimators in turbo-coded BPSK system. An NDA method can be implemented by setting the *a priori* information as 0 in the APPA estimator. And in the DA method we assume all the transmitted data is known. This is an ideal but impractical condition used only for comparison. Moreover, for the purpose of comparison, these S-curves are generated for a particular data stream in the presence of a particular noise sequence with $E_b/N_0 = 2$ dB. The timing offset is assumed to be 0.0. And the signal is sampled twice per symbol. Observe how the slope of the NDA curve is less than the DA, showing that it is more sensitive to noise, and indeed there is an additional estimation error in it. The APPA curves lie between the NDA and the DA. Its result at the 3rd iteration shows an obvious improvement over the second iteration, and it is nearly identical to the DA method. It will tend to DA as more iterations are completed, when the decoding is reliable enough. That is, the accuracy of the timing estimation provided by the APPA improves over the NDA and approaches the DA iteration by iteration.

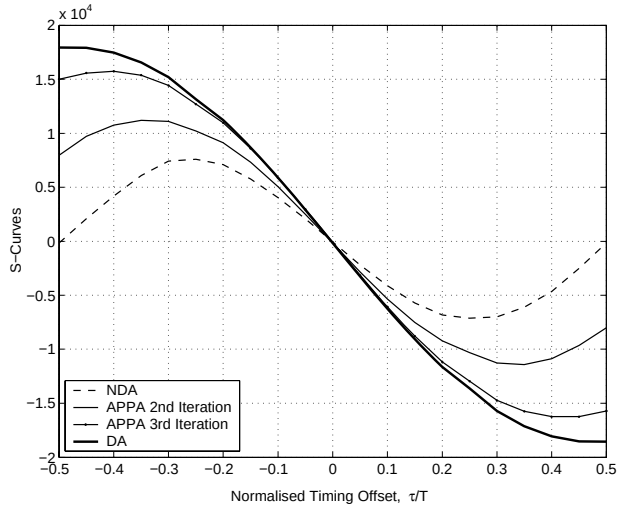
When the system has a non-zero fixed timing offset, we have noted that the zero crossing on the negative slope of the S-curve should be around the true timing offset. This is demonstrated by the simulation results shown in Fig. 1(b). These S-curves are obtained by simulation using the APPA method for the same turbo-coded BPSK system with $E_b/N_0 = 2$ dB and timing offset $\tau = 0.2T$. Different S-curves correspond to different number of iterations. Obviously, the zero crossings on their negative slope are close to $0.2T$ as expected, and the slope at the crossing point increases with more decoding iterations, implying that the accuracy of the timing estimation is improved by performing more iterations.

E. Evaluation of Timing Estimate using Linear Interpolation Algorithm

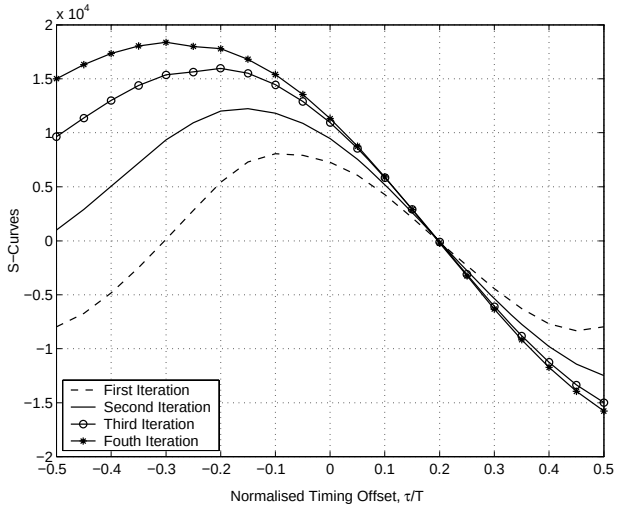
Now the remaining problem is to obtain the ML timing estimate with reasonable complexity. The above analysis of the S-curve suggests that the ML timing estimate can be found from the zero crossing on the negative slope of the S-curve. But it is highly wasteful if we generate the whole S-curve in order to obtain the estimate, since only one sample is desired (the timing offset estimate). A more efficient method is to use a linear interpolation algorithm [11] as depicted in Fig. 2.

This algorithm only calculates four samples at $\{-T/2, -T/6, T/6, T/2\}$ on the S-curve and then chooses two points from these four samples which satisfy the following criterion

$$S(\tau_i) > 0 \quad \text{and} \quad S(\tau_i) \cdot S(\tau_{i+1}) < 0 \tag{15}$$



(a) S-curves for DA, APPA, and NDA Timing Estimators in Turbo-coded BPSK System with $E_b/N_0 = 2$ dB, and Timing Offset=0.0.



(b) S-curves for APPA Timing Estimator in Turbo-coded BPSK System with 4 Different Iterations, $E_b/N_0 = 2$ dB, and Timing Offset=0.2T.

Fig. 1. Illustrating S-curves for Turbo-coded BPSK System.

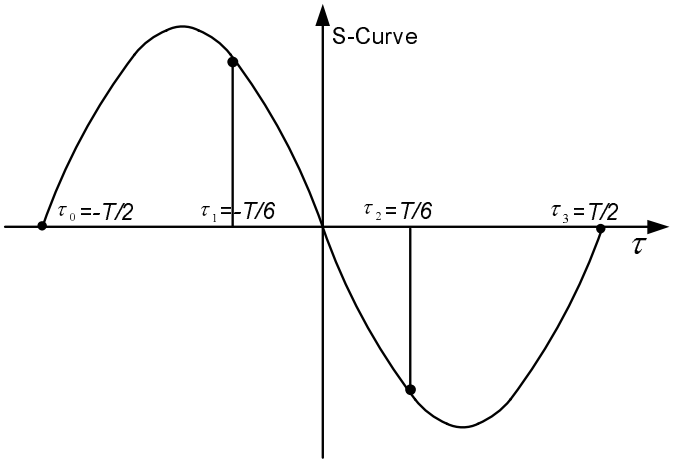


Fig. 2. Four Samples at $\{-T/2, -T/6, T/6, T/2\}$ on the S-curve Used by the Linear Interpolation Algorithm to Determine the ML Timing Estimate

where $S(\tau_i)$ denotes a point on the S-curve with $\tau = \tau_i$. Once τ_i is chosen, the ML estimate $\hat{\tau}_{ML}$ can be found using linear interpolation

$$\hat{\tau}_{ML} = \frac{\tau_i S(\tau_{i+1}) - \tau_{i+1} S(\tau_i)}{S(\tau_{i+1}) - S(\tau_i)}. \tag{16}$$

Noting that the S-curve given by (13) is the sum of the derivative LLF for each individual symbol, thus these four points on the S-curve can be computed for every symbol and summed over one block. The computation of the whole S-curve and the evaluation of the highly complicated equation (13) are avoided. In addition, we pre-calculated the impulse response $g(t+\tau)$ and $g'(t+\tau)$ for the timing offset values of $\{\tau = -T/2, -T/6, T/6, T/2\}$ and stored them into look-up tables to save computational burden.

In summary, the iterative APPA timing estimator works as follows. For every block, at iteration number $l = 0$, timing estimate $\hat{\tau}^0$ and extinsic information $L^0(k)$ are initialised as 0. At the l-th iteration, $l \geq 1$, the derivative of the LLF for four delay values of $\{-T/2, -T/6, T/6, T/2\}$ is computed for each received symbol according to (13) using the extrinsic information obtained at the $(l-1)$-th turbo decoding iteration and the corresponding $g(t+\tau)$ and $g'(t+\tau)$ are obtained from the look-up table. For each delay value, the derivatives of the LLF are summed over one block and this gives the four points on the S-curve for this iteration of this block. Using the linear interpolation algorithm, we obtain the l-th timing estimate, which is used to correct the timing offset in the $(l+1)$-th iteration.

III. Receiver Structure

The block diagram of the receiver using the APPA symbol timing recovery in a turbo coding system is given in Fig. 3. Every component in the receiver operates once for each iteration of every data block. The received signal is fed simultaneously to the matched filter and the timing estimator. The matched filter $g(-t)$ shapes the overall baseband impulse response to satisfy the Nyquist criterion; the function of the interpolation embedded in the matched filter is to generate symbol synchronised samples suitable for detection. For the first iteration, the timing corrector does not make any adjustment, while the turbo-decoder does not provide any information to the synchroniser and the timing estimator is in fact making the estimation using the NDA method. For subsequent iterations, the timing estimator obtains more and more reliable *a priori* information from the decoder and as we will show in the following, this results in a more and more accurate estimate which will be used in the corrector to make adjustment on the data sequence fed to the decoder. This adjustment undoubtedly improves the decoding, which in turn will improve the synchronisation. As long as it proceeds, the reliability of the extrinsic information and the accuracy of the symbol timing estimate improves, until the synchronisation and the decoding converge to correct values after several iterations. Remember, soft demodulation is used before the corrected sequence is fed to the decoder. It is worth mentioning that the system discussed here is a fully digital implementation. Sampling is done using a fixed clock, and thus it is

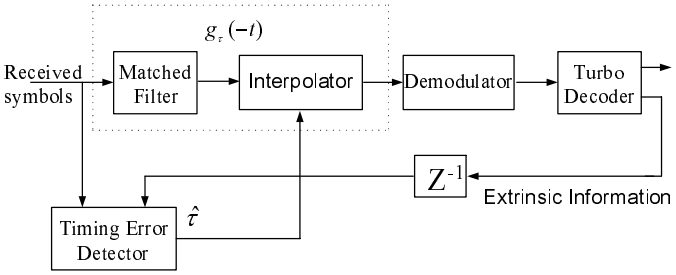


Fig. 3. Block Diagram of a Receiver Using the APPA Timing Recovery in a Turbo Coding System.

impossible to adjust the sampling clock. All timing adjustment must be performed on the signal. In this research, the timing adjustment is implemented by an interpolation filter which is merged with the matched filter to ensure that the interpolator causes negligible distortion in the data processing. As shown in Fig. 3, $g_\tau(-t)$ is the matched filter in the receiver with varying filter coefficients for different timing delay τ. We pre-calculate these coefficients and store them in look-up tables to reduce the computational burden.

IV. Performance

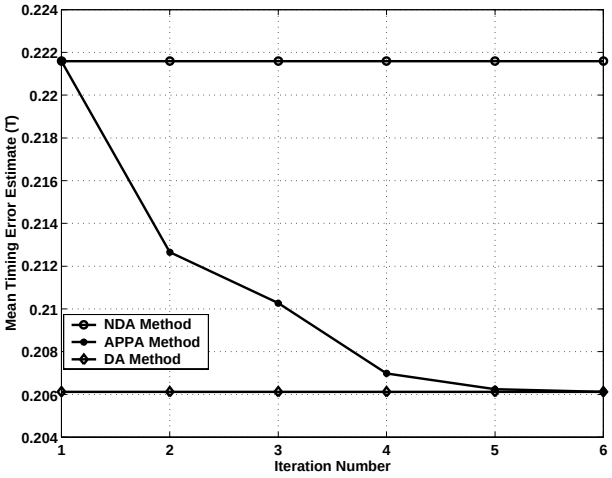
The performance of this algorithm is evaluated in a turbo-coded BPSK system over AWGN channel. With the purpose of comparison, we also implemented NDA and DA timing recovery methods. The NDA method is implemented simply by setting the *a priori* information in the APPA method as 0. And we assume that all the transmitted data is known in the DA method, which is an ideal but unpractical condition used only for comparison.

The 16-state, rate 1/2 turbo code with generator polynomial $G = \{023, 037\}_8$, and length 1024-bit S-random interleaver with an S-parameter of $s = 19$ is used in the simulation. We assume that the timing offset τ remains constant throughout the block and changes only between the blocks. The signalling pulse shape filter $g(t)$ is chosen as a root raised cosine pulse with roll-off factor 1. The signal is sampled twice per symbol (for 100% roll-off filtering this is the minimum required by the sampling theorem, and implies that no information is lost in sampling).

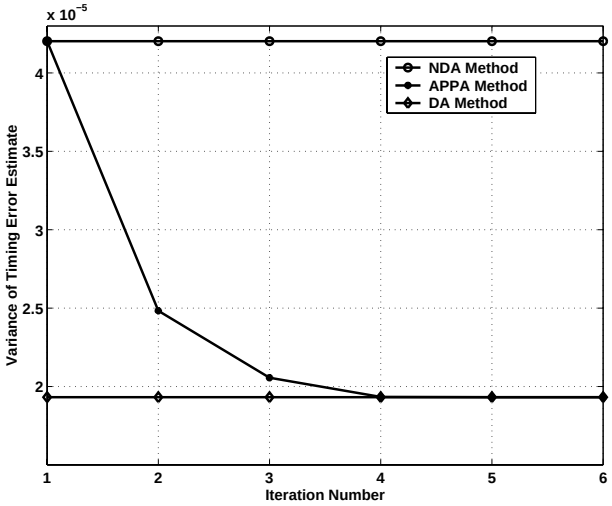
In the following, the performance of the timing recovery algorithm will be assessed through numerical evaluation of the mean of the timing estimate and the variance of the timing estimator. We will also investigate the overall BER performance versus different timing offsets and the energy per bit to noise spectral density ratio E_b/N_0. All the results are obtained by Monte Carlo simulation for 1000 blocks.

A. Compared with the DA and NDA Methods

The difference between these three methods have been demonstrated by their S-curves shown in Fig.1. It can also be identified easily from Fig.4, which illustrates the mean (Fig.4(a)) and the variance (Fig. 4(b)) of the timing estimation for every joint synchronisation and decoding iteration. Fig. 4(a) depicts the mean of the timing estimate for a fixed



(a) Mean Timing Estimate versus Iteration Numbers for NDA, APPA and DA Timing Estimation Methods with $E_b/N_0 = 1.5$ dB and Timing Offset = 0.2T, BPSK Modulation.



(b) Variance of the Timing Estimation versus Iteration Numbers for NDA, APPA and DA Timing Estimation Methods with E_b/N_0 = 2 dB and Timing Offset = 0.2T, BPSK Modulation.

Fig. 4. Comparison with the NDA and DA methods.

timing offset $0.2T$ of the NDA, APPA and DA methods for different iterations with $E_b/N_0 = 1.5$ dB. The number of iterations did not make any difference to the mean estimate of the NDA and DA methods and their curves appear as two straight lines parallel to the x-axis. The curve for APPA lies between them, as we expect. At the first iteration it equals to the NDA method, as no *a priori* information is available. As more iterations are completed, the reliability of the *a priori* information improves, and the estimate generated from the APPA approaches the results obtained from DA, and is superior to NDA. By the 5^{th} iteration (at this E_b/N_0 value), the APPA method matches the DA method.

The same is true for the variance of the timing estimator (i.e. var$\{\hat{\tau} - \tau\}$), as shown in Fig. 4(b). That is the APPA provides a performance starting with that of the NDA and approaching the DA with enough synchronisation/decoding iterations.

B. Variance of Timing Estimator Compared with MCRB

The performance of the APPA timing estimator is also compared to the Modified Cramer-Raw Bound (MCRB) for any unbiased estimator [17]

$$MCRB(\tau) = \frac{T^2}{8\pi^2 \xi N(E_s/N_0)} = \frac{T^2}{4\pi^2 \xi N(E_b/N_0)} \quad (17)$$

where *N* is the number of symbols in the data block and ξ the normalised mean square bandwidth of the pulse spectrum. For the root-raised cosine filter used in this paper, ξ is given by [17]

$$\xi = \frac{1}{12} + \beta^2 \left(\frac{1}{4} - \frac{2}{\pi^2} \right). \quad (18)$$

Note that for 1/2 rate coding BPSK system $E_s = E_b/2$.

Fig. 5 shows the variance of the timing estimator for the NDA and the APPA with the 2^{nd}, 3^{rd} and 6^{th} iterations. It is easy to see that the variance curve for the traditional NDA is much poorer than that of the APPA. The variance of the APPA timing estimator improves and tends to the MCRB as more iterations are completed. With 6 iterations, it reaches a performance floor at some E_b/N_0 value. However, it is noted that the APPA method does not achieves the MCRB although it matches the performance of the DA method after several iterations as shown in Fig. 4(b). This may be caused by the linear interpolation algorithm used to find the ML estimate or the interpolation algorithm used for the timing correction. It is also noted that the curve with 6 iterations is inferior to that with 2 iterations in the region of $E_b/N_0 < 1$ dB. This is because that at such low E_b/N_0, the decoding is not reliable even with ideal synchronisation. With more decoding iterations, only a proportion of blocks are improved. The same happens to the timing estimates generated from these blocks, relying on the extrinsic information obtained from the turbo decoder. This leads to more variable estimates, which consequently results in the increase of the variance. If the decoding converges with enough iterations, when all the blocks have the same performance, then the variable range of timing estimates, or equivalently the variance will reduce accordingly.

C. BER Performance

In Fig. 6, we show the overall BER performance of a turbo-coded BPSK system against symbol timing offset using the APPA symbol timing recovery approach. These curves were generated by computer simulation with four synchronisation/decoding iterations. Two curves correspond to two E_b/N_0 values: 1.25 dB and 1.5 dB. It is shown that the overall BER performance is kept nearly unchanged up to 0.45T, meaning that the system can provide reliable performance in the presence of a timing offset in the region of $[-0.45T, 0.45T]$. Note that random variations are visible on the curve of 1.25 dB as a result of the Monte Carlo simulation used to obtain the results. These could be reduced by increasing the number of random trials.

The overall BER performance versus E_b/N_0 values is shown in Fig. 7 with a fixed timing offset $0.2T$ in comparison with the ideal curve and the BER degradation curve resulting from this timing offset without performing any timing recovery. It is shown that the APPA curve after six iterations exhibits a negligible performance degradation with respect to that with ideal synchronisation. Although the BER performance of the conventional NDA method has not been included in this figure, it has been shown in Mielczarek's work [6] that the ML symbol timing recovery algorithm reaches a performance floor in a turbo-coded system and the performance is much worse than their proposed method, which is about 0.1-0.2 dB away from the ideal performance curve.

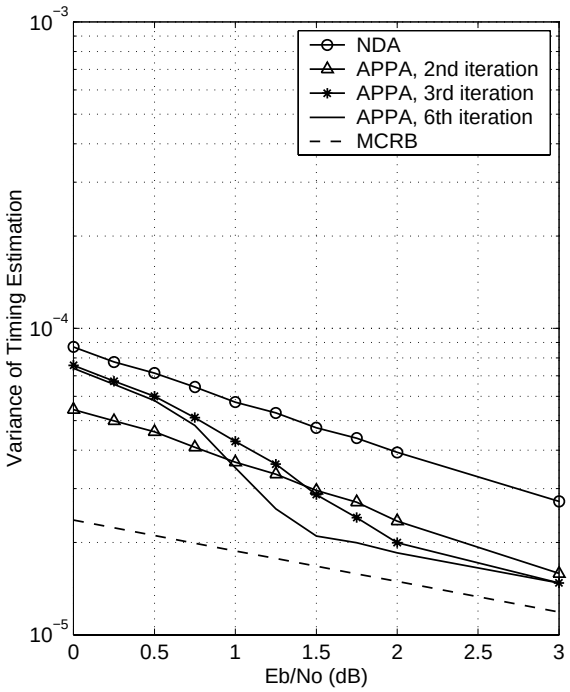


Fig. 5. Variance of Timing Estimation versus E_b/N_0 using NDA and APPA Timing Estimation Methods Compared with the MCRB, Timing Offset = 0.2T, BPSK Modulation.

V. Conclusion

In this paper, we have presented the APPA symbol timing recovery algorithm, which is based on the Maximum Likelihood Estimation strategy with the aid of the *extrinsic* information generated in the turbo decoding iteration. This information is used to improve the iterative decoding as well as the synchronisation through a joint log-likelihood function. This method has two distinctive characteristics in comparison with the traditional timing synchronisation methods, NDA and DA: firstly it operates iteratively; secondly, it is combined with the turbo decoding through iteratively using the extrinsic information generated by the turbo decoder, rather than prior to the decoder separately like in the NDA and DA methods. The most likely estimate is obtained by working out the zero crossing of the negative slope of the S-curve using a linear interpolation algorithm based on four samples taken on the S-curve without the necessity to generate the whole S-curve, and thus significantly reduce the computational load.

The system performance was assessed through the mean of the timing estimate, the variance of the timing estimation

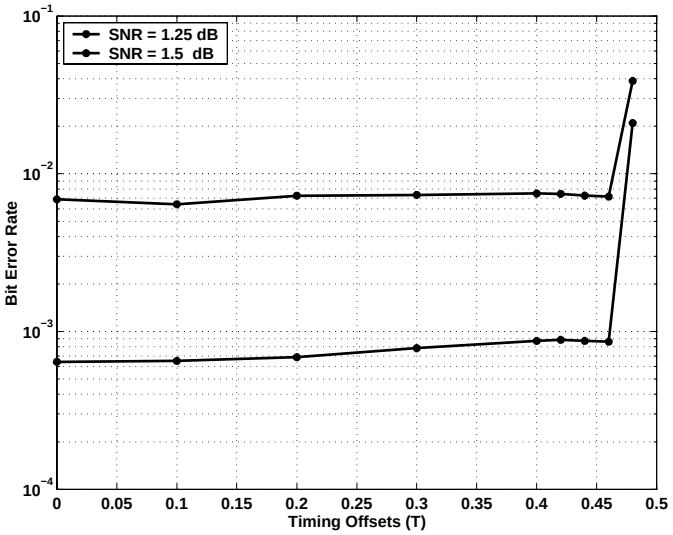


Fig. 6. BER Performance versus Timing Offsets using APPA Timing Estimation Method for Turbo-coded BPSK System after Four Iterations for E_b/N_0=1.25 dB and 1.5 dB.

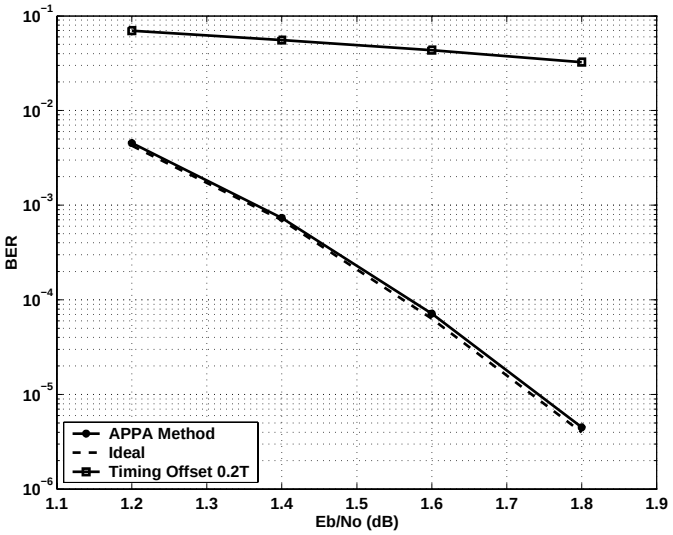


Fig. 7. BER Performance versus E_b/N_0 using APPA Timing Estimation Method for Turbo-coded BPSK System after Six Iterations with Timing Offset 0.2T.

and the overall BER performance. It was shown by these simulation results that the APPA algorithm provides very good system performance with a great improvement over that of the NDA, and approaches the DA without the need for employing pilot symbols.

Iterative joint decoding and synchronisation using APPA estimation can be applied in many other cases. Most straightforwardly it can be used with other forms of modulation, such as QPSK and 8PSK, or in a coded system with unknown carrier phase and symbol timing offset. The research results of these applications will be reported in future work.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo codes," in *The IEEE International Conference of Communications*, pp. 1064–1070, 1993.

[2] A. Burr, *Modulation and Coding for Wireless Communications*. Prentice Hall, 2001.

[3] L. Zhang and A. Burr, "Iterative carrier phase recovery suited to turbo coded systems," *IEEE Transaction on Wireless Communications*, vol. 11, 2004.

[4] L. Lu and S. G. Wilson, "Synchronisation of turbo coded modulation systems at low SNR," in *International Conference on Communications*, vol. 1, pp. 428–432, June 1998.

[5] J. G. Proakis, *Digital Communications.* McGRAW-HILL International Editions, 1995.

[6] B. Mielczarek and A. Svensson, "Timing error recovery in turbo coded systems on AWGN channel," *IEEE Transaction On Communication*, vol. 50, no. 10, pp. 1584–1592, 2002.

[7] B. Mielczarek and A. Svensson, "Joint synchronisation and decoding of turbo codes on AWGN channels," in *IEEE Vehicular Technology Conference*, vol. 1, pp. 1886–1890, 1999.

[8] J. Barry, A. Kavcic, S. McLaughlin, A. Nayak, and W. Zeng, "Iterative timing recovery," *IEEE Signal Processing Magazine*, January 2004.

[9] H. L. V. Trees, *Detection, Estimation, and Modulation Theory*, vol. 1. Wiley, 1968.

[10] L. E. Franks, "Carrier and bit synchronisation in data communication–a tutorial review," *IEEE Transactions On Communications*, vol. COM-28, pp. 1107–1120, 1980.

[11] D. O'Shea, "A novel interpolation method for maximum likelihood timing recovery," *Wireless Personal Communications*, vol. 4, pp. 315–324, 1997.

[12] H. Meyr, M. Moeneclaey, and S. A. Fechtel, *Digital Communication Receviers: Synchronisation, Channel Estimation and Signal Processing*. Wiley Series in Telecommunicationa and Signal Processing, John Wiley and Sons, Inc, 1997.

[13] F. M. Gardner, "Timing adjustment via interpolation in digital demodulators," final report: part I, Gardner Research Company, 1755 University Avenue, Palo Alto, California 94301 (USA), June 1990.

[14] A. Burr, "Turbo-codes: the ultimate error control codes?," *Electronics and Communicaiton Engineering Journal*, pp. 155–165, August 2001.

[15] G. Colavolpe, G. Ferrari, and R. Raheli, "Extrinsic information in iterative decoding: A unified view," *IEEE Transaction On Communication*, vol. 49, No. 12, 2001.

[16] L. E. Franks, *Digital Communications: Satellite/Earth Station Engineering*, ch. Synchronisation Subsystems: Analysis and Design, pp. 294–317. Prentice-Hall, k. feher ed., 1983.

[17] U. Mengali and A. N. D'Andrea, *Synchronisation Techniques for Digital Receivers*. Plenum Press, 1997.

Li Zhang received the BSc degree in Electronic Engineering in 1995 and the MEng degree in Signal and Information Processing in 1998 from the Department of Electronics, Beijing University of Aeronautics and Astronautics, Beijing, China. She received her Ph.D degree in communications from the University of York, York, UK in 2003.

From 1999 to 2000, she worked as a research scientist at China Software Design Centre, Agilent Technologies Ltd., Beijing, China. From 2000 to 2003, she was a PhD research student in the Communications Research Group, University of York, U.K. She is currently a lecturer in communications with the University of Leeds. Her research interests lie in the areas of wireless communications and signal processing, including synchronisation, coding and modulation, turbo-codes, MIMO systems, space-time coding, OFDM etc.

Alister Burr was born in London, U.K, in 1957. He received the BSc degree in Electronic Engineering from the University of Southampton, U.K in 1979 and the PhD from the University of Bristol in 1984. Between 1975 and 1985 he worked at Thorn-EMI Central Research Laboratories in London. In 1985 he joined the Department of Electronics at the University of York, U.K, where he has been Professor of Communications since 2000. His research interests are in wireless communication systems, especially modulation and coding and including turbo-codes and turbo-processing techniques, MIMO systems and Space-Time coding. In 1999 he was awarded a Senior Research Fellowship by the U.K. Royal Society, and in 2002 he received the J. Langham Thompson Premium from the Institution of Electrical Engineers. He has also held a visiting professorship at Vienna University of Technology. He is currently chair, working group 1, of the European COST 273 programme ”Towards Broadband Mobile Networks”.