RADIAL BASIS FUNCTION ASSISTED TURBO EQUALISATION

M. S. Yee, B. L. Yeap, L. Hanzo

Dept. of ECS, Univ. of Southampton, SO17 1BJ, UK.
Tel: +44-703-593 125, Fax: +44-703-594 508
Email:lh@ecs.soton.ac.uk; http://www-mobile.ecs.soton.ac.uk

ABSTRACT

This paper presents a novel turbo equalisation scheme, which employs a Radial Basis Function (RBF) Decision Feedback Equaliser (DFE) and the so-called Jacobian logarithmic complexity reduction technique instead of the conventional trellis-based equaliser. The proposed turbo equaliser is shown to achieve identical bit error rate (BER) performance to the conventional turbo equaliser, while incurring a factor 4.4 lower 'per-iteration' complexity in the context of 4-level Quadrature Amplitude Modulation (4QAM).

1. INTRODUCTION

In conventional systems equalisation and channel decoding ensues independently. However, it is possible to improve the receiver's performance, if the equaliser is fed by the channel outputs plus the soft decisions provided by the channel decoder, invoking a number of iterative processing steps. This new receiver scheme, which was first proposed by Douillard *et al.* [1] for a serially concatenated convolutional coded binary phase shift keying (BPSK) system, uses the same principle as turbo codes and hence it was termed *turbo-equalisation.* This scheme is illustrated in Figure 1.

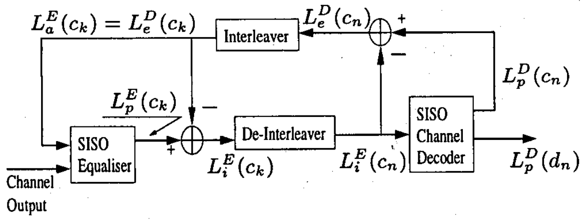

Figure 1: Iterative turbo equalisation schematic

The principles of iterative turbo decoding [2] were modified for the serially concatenated system of Figure 2. The channel encoder is fed with independent binary data d_n and every $\log_2(\mathcal{M})$ number of bits of the interleaved, channel encoded data c_k is mapped to an $\mathcal{M}$-ary symbol before transmission. In this scheme the channel is viewed as an 'inner encoder' of a serially concatenated arrangement. At the receiver the equaliser and decoder employ a Soft-In/Soft-Out (SISO) algorithm, such as the optimal Maximum *A Posteriori* MAP algorithm [3] or the Log-MAP algorithm [4]. The SISO equaliser/decoder processes the *a priori* information associated with the coded bits c_k transmitted over the channel and – in conjunction with the channel output values v_k – computes the *a posteriori* information concerning the coded bits. The soft value of a channel coded bit c_k is typically quantified in the form of the Log Likelihood Ratio (LLR) $L(c_k)$, which is defined as [1]:

$$L(c_k) \triangleq \ln\left(\frac{P(c_k=+1)}{P(c_k=-1)}\right). \qquad (1)$$

Note that in the context of turbo decoding the SISO decoders compute the *a posteriori* information of the *source* bits, while in turbo equalisation the *a posteriori* information concerning the *coded* bits is also required.

In our description of the turbo equaliser depicted in Figure 1, we have used the notation L^E and L^D to indicate the LLR values output by the SISO equaliser and SISO decoder, respectively. The subscripts e, i, a and p were used to represent the extrinsic LLR, the combined channel and extrinsic LLR, the *a priori* LLR and the *a posteriori* LLR, respectively. Referring to Figure 1, the SISO equaliser accepts the channel outputs and the *a priori* information of the coded bits $L_a^E(c_k)$, and yields the *a posteriori* LLR values $L_p^E(c_k)$ of the coded bits c_k. Before passing the above *a posteriori* LLRs generated by the SISO equaliser to the SISO decoder of Figure 1, the contribution of the decoder — in the form of the *a priori* information $L_a^E(c_k)$ — from the previous iteration must be removed, in order to yield the combined channel and extrinsic information $L_i^E(c_k)$ seen in Figure 1. They are refered to as 'combined', since they are intrinsically linked and cannot be separated. However, note that at the initial iteration stage, no *a priori* information is available yet, hence we have $L_a^E(c_k) = 0$. The *a priori* information $L_a^E(c_k)$ was removed at this stage, in order to avoid that the decoder processes its own output information, which would result in overwhelming the decoder's current reliability-estimation of the coded bits, *i.e.* the extrinsic information. The combined channel and extrinsic LLR values are channel-deinterleaved – as seen in Figure 1 – to yield $L_i^E(c_n)$, which is then passed to the SISO channel decoder. Subsequently, the channel decoder computes the *a posteriori* LLR values of the coded bits $L_p^D(c_n)$. The *a posteriori* LLRs at the output of the channel decoder are constituted by the extrinsic LLR $L_e^D(c_n)$ and the channel-deinterleaved combined channel and extrinsic LLR $L_i^E(c_n)$ extracted from the equaliser's *a posteriori* LLR $L_p^E(c_k)$.

VTC'2000 TOKYO, JAPAN, 15-18 MAY 2000

The financial support of the following organisations is gratefully acknowledged: EPSRC, UK, the Comission of the European Communities, Brussels, Belgium and Motorola ECID, Swindon, UK and the Virtual Centre of Excellence (VCE) in Mobile Communications.

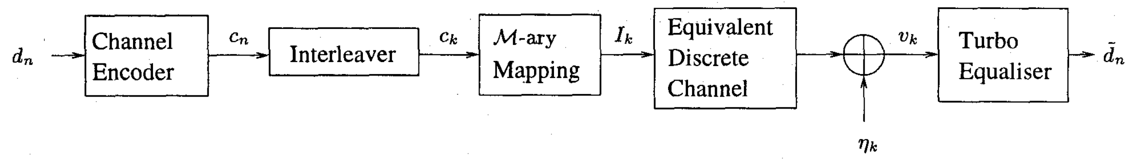

Figure 2: Serially concatenated coded $\mathcal{M}$-ary system using the turbo equaliser, which performs the equalisation, demodulation and channel decoding iteratively.

The extrinsic part is the incremental information about the current bit obtained through the decoding process from all the information available for the other bits, but excluding the information directly conveyed by the bit. This information can be calculated by subtracting bitwise the LLR values $L_i^E(c_n)$ at the input of the decoder from the *a posteriori* LLR values $L_p^D(c_n)$ at the channel decoder's output, as seen also in Figure 1:

$$L_e^D(c_n) = L_p^D(c_n) - L_i^E(c_n). \tag{2}$$

The extrinsic information $L_e^D(c_n)$ of the coded bits is then interleaved in Figure 1, in order to yield $L_e^D(c_k)$, which is fed back to the equaliser, where it is used as the *a priori* information $L_a^E(c_k)$ in the next equalisation iteration. This constitutes the first iteration. It is important that only the channel-interleaved extrinsic part - *i.e.* $L_e^D(c_k)$ of $L_p^D(c_n)$ - is fed back to the equaliser, since the correlation between the *a priori* information $L_a^E(c_k) = L_e^D(c_k)$ used by the equaliser and previous decisions of the equaliser should be minimized. This is to obtain the equaliser's reliability estimation of the coded bits for the current iteration, without being 'influenced' by its previous estimations. Ideally, the *a priori* information should be based on an independent estimation. As argued above, this is the reason that the *a priori* information $L_a^E(c_k)$ is subtracted from the *a posteriori* LLR value $L_p^E(c_k)$ at the output of the equaliser in Figure 1, before passing the LLR values to the channel decoder. In the final iteration, the *a posteriori* LLRs $L_p^D(d_n)$ of the source bits are computed by the channel decoder. Subsequently, the transmitted bits are determined by comparing $L_p^D(d_n)$ to the threshold value of 0. For $L_p^D(d_n) < 0$ the transmitted bit d_n is deemed to be a logical 0, while $d_n = +1$ is output, when $L_p^D(d_n) \geq 0$.

Previous turbo equalistion research has implemented the SISO equaliser using the Soft-Output Viterbi Algorithm SOVA in [1], the optimal MAP algorithm in [5] and linear filters in [6]. We will now introduce the proposed Radial Basis Function (RBF) based equaliser as the SISO equaliser in the context of turbo equalisation. The following sections will discuss the implementational details and the performance of this scheme, benchmarked against the optimal MAP turbo equaliser scheme of [5].

2. RBF ASSISTED TURBO EQUALISATION

The RBF network based equaliser is capable of utilizing the *a priori* information $L_a^E(c_k)$ provided by the channel decoder of Figure 1, in order to improve its performance. This *a priori* information can be assigned namely as the weights of the RBF network [7]. We will desribe this in more detail in this section.

The conditional probability density function (PDF) of the ith symbol, $i = 1, \ldots, \mathcal{M}$, associated with the ith subnet of the $\mathcal{M}$-ary RBF equaliser is given by [7]:

$$\begin{aligned} f_{RBF}^i(\mathbf{v}_k) &= \sum_{j=1}^{n_s^i} w_j^i \varphi(|\mathbf{v}_k - \mathbf{c}_j^i|), & (3)\\ w_j^i &= p_j^i (2\pi\sigma_\eta^2)^{-m/2}, & (4)\\ \varphi(x) &= \exp\left(\frac{-x^2}{2\sigma_\eta^2}\right) & (5)\\ & i = 1,\ldots,\mathcal{M}, \qquad j = 1,\ldots,n_s^i \end{aligned}$$

where $\mathbf{c}_j^i$, $\mathbf{w}_j^i$ and $\varphi(\cdot)$ are the RBF's centers, weights and activation function, respectively, and σ_η^2 is the noise variance of the channel. The size of the vectors $\mathbf{v}_k$ and $\mathbf{c}_j^i$ is equivalent to the equaliser order m. The term p_j^i is the probability of occurance of the channel state $\mathbf{r}_j^i$ and it determines the values of the RBF weights w_j^i. The actual number of channel states n_s^i is determined by the design of the algorithm that reduces the number of channel states from the optimum number of $\mathcal{M}^{m+L-1}$ [8, 9, 10]. The probability of the channel states $\mathbf{r}_j^i$ and therefore the weights of the RBF equaliser can be derived from the LLR values of the transmitted bits, as estimated by the channel decoder. The channel output state, which is the product of the CIR matrix $\mathbf{F}$ and the channel input state $\mathbf{s}_j$, is represented as follows [7]: $\mathbf{r}_j = \mathbf{F}\mathbf{s}_j$, where the channel impulse response (CIR) of length $L+1$ is represented by $F(z) = \sum_{n=0}^{L} f_n z^{-n}$ and $\mathbf{F}$ is an $m \times (m+L)$ matrix given by the CIR taps as follows:

$$\mathbf{F} = \begin{bmatrix} f_0 & f_1 & \cdots & f_L & \cdots & 0 \\ 0 & f_0 & \cdots & f_{L-1} & \cdots & 0 \\ \vdots & \vdots & & & & \vdots \\ 0 & 0 & f_0 & \cdots & f_{L-1} & f_L \end{bmatrix}. \tag{6}$$

The channel input state $\mathbf{s}_j$ is given by the jth combination of $(L+m)$ possible transmitted symbols, namely by $\mathbf{s}_j = \begin{bmatrix} s_{j1} & \ldots & s_{jl} & \ldots & s_{j(L+m)} \end{bmatrix}^T$. Hence - for a time-invariant CIR - the probability of the received channel output vector is given by

$$\begin{aligned} p(\mathbf{r}_j) &= p(\mathbf{s}_j) \\ &= p(s_{j1} \cap \ldots s_{jl} \cap \ldots s_{j(L+m)}) \\ &= p(s_{j1}) \cdot \ldots p(s_{jl}) \cdot \ldots p(s_{j(L+m)}) \\ & \qquad j = 1, \ldots, n_s^i. \end{aligned} \tag{7}$$

The transmitted symbol vector component s_{jl} – i.e. the lth symbol in the jth vector, $j = 1 \ldots n_s^i$ – is represented by

$\mathrm{m} = \log_2 \mathcal{M}$ number of bits $c_{jl1}, c_{jl2}, \ldots, c_{jlm}$. Therefore,

$$\begin{aligned} p(s_{jl}) &= p(c_{jl1} \cap \ldots c_{jlm} \cap \ldots c_{jl\mathrm{m}}) \\ &= p(c_{jl1}) \cdot \ldots p(c_{jlm}) \cdot \ldots p(c_{jl\mathrm{m}}) \\ & \quad j = 1 \ldots n_s^i, \qquad l = 1, \ldots, L + m. \end{aligned} \tag{8}$$

We have to map the bits c_{jlm} representing the $\mathcal{M}$-ary symbol s_{jk} to the corresponding bit c_k. The probability of the bit c_k being logical 0 or 1 can be obtained from the *a priori* information $L_a^E(c_k)$ provided by the channel decoder of Figure 1, defined in Equation 1 according to:

$$P(c_k = \pm 1) = \frac{\exp(-\mathcal{L}(c_k)/2)}{1 + \exp(-\mathcal{L}(c_k))} \cdot \exp(\pm \mathcal{L}(c_k)/2). \tag{9}$$

Therefore, we have demonstrated how the soft output $L_a^E(c_k)$ of the decoder can be utilized by the RBF equaliser. Another way of viewing this process is that the RBF equaliser is trained by the information generated by the decoder. The RBF equaliser provides the *a posteriori* LLR values of the bit c_k according to

$$\mathcal{L}_p(c_k) = \ln \left(\frac{\sum_{c_k=+1}^{i} f_{RBF}^i(\mathbf{v}_k)}{\sum_{c_k=-1}^{i} f_{RBF}^i(\mathbf{v}_k)} \right). \tag{10}$$

In the next section we will provide a comparative study of the RBF equaliser with the MAP equaliser.

3. COMPARISON OF THE RBF AND MAP EQUALISER

The *a posteriori* LLR value of the coded bit c_k can be calculated according to [5]:

$$\mathcal{L}_p(c_k) = \ln \left(\frac{\sum_{\substack{(s',s) \\ c_k=+1}} p(s', s, v_k)}{\sum_{\substack{(s',s) \\ c_k=-1}} p(s', s, v_k)} \right), \tag{11}$$

where s' and s denote the states of the trellis at trellis stages $k-1$ and k, respectively. The joint probability $p(s', s, \mathbf{v}_k)$ is the product of three factors [5]:

$$p(s', s, \mathbf{v}) = \underbrace{p(s', v_{j<k})}_{\alpha_{k-1}(s')} \cdot \underbrace{P(s|s') \cdot p(v_k|s', s)}_{\gamma_k(s',s)} \cdot \underbrace{p(v_{j>k}|s)}_{\beta_k(s)}, \tag{12}$$

where the term $\alpha_{k-1}(s')$ and $\beta_k(s)$ are the forward- and backward oriented transition probabilities, respectively, which can be obtained recursively, as follows [5]:

$$\begin{aligned} \alpha_k(s) &= \sum_{s'} \gamma_k(s', s) \cdot \alpha_{k-1}(s') & (13) \\ \beta_{k-1}(s) &= \sum_{s} \gamma_k(s', s) \cdot \beta_k(s). & (14) \end{aligned}$$

Furthermore, $\gamma_k(s', s), k = 1, \ldots, \mathcal{F}$ represents the trellis transitions between the states $(k-1)$ and k. The trellis has to be of finite length and for the case of MAP equalisation, this corresponds to the length of the received transmission burst $\mathcal{F}$. The branch transition probability $\gamma_k(s', s)$ can be expressed as the product of the *a priori* probability $P(c_k)$ and the transition probability $p(v_k|s', s)$:

$$\gamma_k(s', s) = p(v_k|s', s) \cdot P(c_k). \tag{15}$$

The transition probability is calculated according to [5]

$$\begin{aligned} \gamma_k(s', s) &= \gamma_k^*(s', s) \cdot \exp(\frac{1}{2} \cdot c_k \cdot \mathcal{L}(c_k)), & (16) \\ \gamma^*(s', s) &= \exp(-\frac{|v_k - \tilde{v}_k|^2}{2\sigma_\eta^2}). & (17) \end{aligned}$$

Note the similarity of the transition probability of Equation 16 with the PDF of the RBF equaliser's ith symbol described by Equation 3, where the terms $\exp(\frac{1}{2} \cdot c_k \cdot \mathcal{L}(c_k))$ and $\gamma^*(s', s)$ are the RBF's weight and activation function, respectively, while the number of RBF nodes n_s^i is one. We also note that the computational complexity of both the MAP and the RBF equalisers can be reduced by representing the output of the equalisers in the logarithmic domain and utilizing the Jacobian logarithmic relationship [11, 12]. The RBF equaliser based on the Jacobian logarithm was hence termed as the Jacobian RBF equaliser in [12]. The computational complexity associated with generating the *a posteriori* LLRs using the Jacobian logarithmic relationship for the Log-MAP equaliser and the Jacobian RBF equaliser is given in Table 1. Due to lack of space, here we refrain from elaborating on the justification of the complexity summarised in Table 1, but we will evaluate the approximate complexity of our schemes investigated in Section 5.

	Log-MAP	Jacobian RBF
subtract and add	$n_{s,f}(6\mathcal{M} + 2) - 3$	$n_{s,f} + \mathcal{M} n_s^i (m + 2) - 4$
multiplication	$n_{s,f}$	$n_{s,f}$
division	$n_{s,f}$	$n_{s,f}$
max	$n_{s,f}(2\mathcal{M} - 1) - 2$	$\mathcal{M} n_s^i - 2$
table look-up	$n_{s,f}(2\mathcal{M} - 1) - 2$	$\mathcal{M} n_s^i - 2$

Table 1: Computational complexity of generating the *a posteriori* LLR for the Log-MAP equaliser and the Jacobian RBF equaliser [12]. The RBF equaliser order is denoted by m and the number of RBF nodes is n_s^i. The notation $n_{s,f} = \mathcal{M}^{L+1}$ indicates the number of trellis states for the Log-MAP equaliser and also the number of scalar channel states for the Jacobian RBF equaliser.

The memory of the MAP equaliser is limited by the length of the trellis, provided that decisions about the symbol I_k, $k = -\infty \ldots \infty$ are made in possession of the information on all the received symbols. The recursions of Equation 13 and 14 in the MAP algorithm avoid the storage of the entire sequence of symbols. The equaliser's delay facilitates invoking information from 'future' samples, $v_k, \ldots, v_{k-\tau+1}$, to be used in the detection of the transmitted symbol $I_{k-\tau}$. The delayed decision of the MAP equaliser provides the necessary information concerning the 'future' samples – relative to the delayed decision – to be utilised and the information of the future samples is generated by the backward recursion of Equation 14.

The MAP equaliser exhibits optimum performance. However, if decision feedback is used in the RBF subset center selection as in [8], the performance of the RBF decision feedback equaliser (DFE) in conjunction with *correct* decision feedback is better, than that of the MAP equaliser due to the increased Euclidean distance between channel states, as it will be demonstrated in Section 5. In the remainder of this paper we use the Jacobian RBF DFE [12] in our proposed turbo equalisation scheme and study its performance in comparison to the MAP equaliser.

4. SYSTEM OVERVIEW

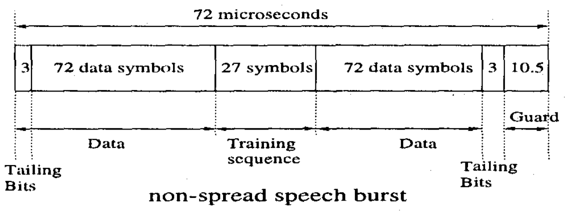

Figure 3: Transmission burst structure of the FMA1 non-spread speech burst of the FRAMES proposal [13]

The schematic of the entire system was shown in Figure 2, where the transmitted source bits are convolutionally encoded, channel-interleaved and mapped to an $\mathcal{M}$-ary modulated symbol. The encoder utilized a half-rate recursive systematic convolutional (RSC) code, having a constraint length of $K = 5$ and octal generator polynomials of $G_0 = 35$ and $G_1 = 23$. A random channel interleaver of 20000-bit memory was invoked. The transmission burst structure used in this system is the FMA1 non-spread speech burst, as specified in the Pan-European FRAMES proposal [13], which is seen in Figure 3. We have assumed that perfect knowledge of the CIR was available and used the Jacobian RBF DFE instead of the trellis-based MAP equaliser.

5. PERFORMANCE RESULTS

The performance of the Jacobian RBF DFE turbo-equaliser (TEQ) was initially investigated over a dispersive Additive White Gaussian Noise (AWGN) channel having a z-domain transfer function of $F(z) = 0.5773 + 0.5773z^{-1} + 0.5773z^{-2}$. Figure 4 provides the BER performance comparison of the Log-MAP and Jacobian RBF DFE based TEQ scheme over the above AWGN channel for BPSK. The Jacobian RBF DFE had a feedforward order of $m = 4$, feedback order of $n = 2$ and decision delay of $\tau = 3$ symbols. Figure 4 shows that when the feedback information is not error-free, the Log-MAP TEQ performs better, than the Jacobian RBF DFE TEQ for the same number of iterations. The corresponding uncoded systems using the Log-MAP equaliser and the Jacobian RBF DFE exhibit similar performance trends. However, both equalisers converge to a similar BER performance upon increasing the number of iterations. The performance of the Log-MAP TEQ in the zero-ISI Gaussian channel environment was also presented in Figure 4 for comparison. Again, the BER performance of the RBF DFE TEQ with *correct* decision fedback shown in Figure 4 exhibits a better performance compared to the Log-MAP TEQ. This is possible – although the Log-MAP equaliser is known to approximate the optimal performance – because the RBF DFE's subset center selection mechanism creates an increased Euclidean distance between the channel states [8], which improves the performance of the Jacobian RBF DFE TEQ.

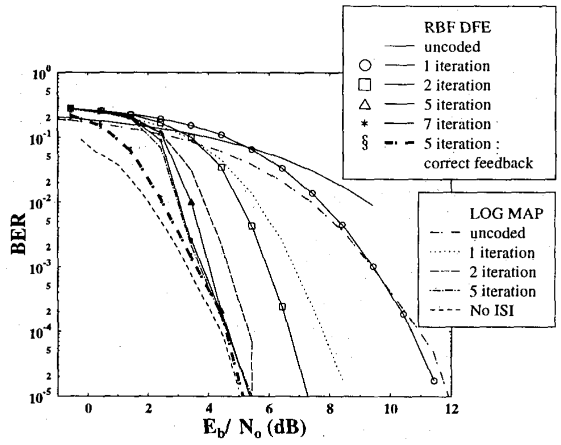

Figure 4: Performance of the Log-MAP TEQ and Jacobian RBF DFE TEQ over three-path AWGN channels for BPSK. The Jacobian RBF DFE has a feedforward order of $m = 4$, feedback order of $n = 2$ and decision delay of $\tau = 3$ symbols.

Let us now investigate the performance of the TEQs in a dispersive Rayleigh fading channel environment. A three-path, symbol-spaced fading channel of equal weights was utilized, where the Rayleigh fading statistics obeyed a normalised Doppler frequency of 1.5×10^{-4}. The CIR was assumed to be frame-invariant. Figure 5 and Figure 6 portray the performance of the Log-MAP TEQ and the Jacobian RBF DFE TEQ for BPSK and 4QAM, respectively. The Jacobian RBF DFE has a feedforward order of $m = 3$, feedback order of $n = 2$ and decision delay of $\tau = 2$ symbols. These figures show that the Log-MAP TEQ and the Jacobian RBF DFE TEQ converge to a similar BER performance, but the Log-MAP TEQ requires a lower number of iterations. Specifically, two iterations are required for the Log MAP TEQ and three iterations for the Jacobian RBF DFE TEQ to achieve near-perfect convergence, since the Log-MAP TEQ exhibited a better BER performance than the Jacobian RBF DFE for an uncoded system. The performance of the Log-MAP TEQ at two iterations and that of the Jacobian RBF DFE TEQ at three iterations is about 2dB and 2.5dB from the zero-ISI Gaussian BER curve for BPSK and 4QAM, respectively, at a BER of 10^{-4}. Since the computation of the associated implementational complexity in Table 1 is quite elaborate, here we only give an estimate of the Log-MAP TEQ's and the Jacobian RBF DFE TEQ's complexity in the context of both BPSK and 4QAM, employing the parameters used in our simulations. Specifically, in the BPSK scheme the approximate number of additions/subtrations and multiplications/divisions for the Log-MAP TEQ was 109 and 16 per iteration, respectively, whereas for the Jacobian RBF DFE TEQ ($m = 3$, $n = 2$, $\tau = 2$) the corresponding figures were 44 and 16, respectively. The 'per iteration' complexity of the Jacobian RBF DFE TEQ was approximately a factor of 2.5 and 4.4 lower, than that of the Log-MAP TEQ, for BPSK and 4QAM, respectively.

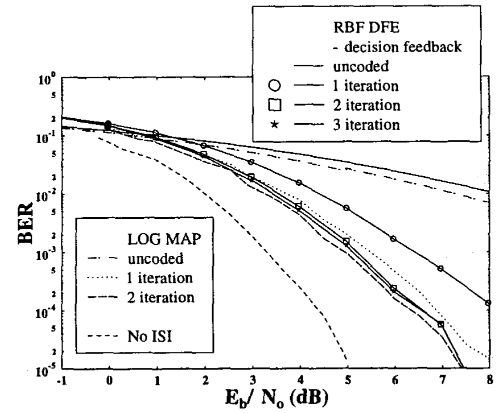

Figure 5: Performance of the Log-MAP TEQ and Jacobian RBF DFE TEQ over three-path Rayleigh fading channels for **BPSK**. The Jacobian RBF DFE has a feedforward order of $m = 3$, feedback order of $n = 2$ and decision delay of $\tau = 2$ symbols.

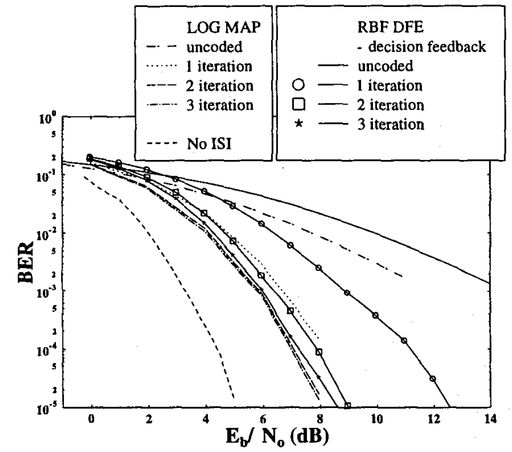

Figure 6: Performance of the Log-MAP TEQ and Jacobian RBF DFE TEQ over three-path Rayleigh fading channels for **4QAM**. The Jacobian RBF DFE has a feedforward order of $m = 3$, feedback order of $n = 2$ and decision delay of $\tau = 2$ symbols.

6. CONCLUSIONS

The Jacobian RBF DFE TEQ has been proposed and analysed comparatively in conjunction with the well-known Log-MAP TEQ. The associated performances and complexities have been compared in the context of both BPSK and 4QAM. It was found that the two schemes exhibit similar performances in all investigated scenarios, and over dispersive Gaussian channels near-narrowband performance was attained by both scheme. Over dispersive Rayleigh fading channels the performances were approximately 2dB away form the non-dispersive AWGN performance bound. The associated 'per iteration' implementational complexity of the Jacobian RBF DFE TEQ was approximately a factor 2.5 and 4.4 lower in the context of BPSK and 4QAM, respectively. Our future work is likely to consider the employment of space-time coding and adaptive beam-steering in the context of the proposed transceiver.

7. REFERENCES

[1] C. Douillard, M. Jézéquel, C. Berrou, "Iterative Correction of Intersymbol Interference: Turbo-Equalization" *European Trans. on Telecommunication*, vol. 6, no. 5, pp. 507–511, September/October 1995.

[2] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: turbo-codes" *IEEE Trans. on Communications*, vol. 44, no 10, pp. 1261-1271, October 1996.

[3] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal Decoding of Linear Codes for Minimising Symbol Error Rate," *IEEE Trans. on Information Theory*, pp. 284–287, March 1974.

[4] P. Robertson, E. Villebrun and P. Hoeher, "A Comparison of Optimal and Sub-Optimal MAP Decoding Algorithms Operating in the Log Domain," *Proc. of the International Conference on Communications, Seattle, USA*, pp. 1009–1013, June 1995.

[5] G. Bauch and H. Khorram and J. Hagenauer, "Iterative Equalization and Decoding in Mobile Communications System" *European Personal Mobile Communications Conference, Bonn, Germany*, pp. 301-312, 30 September – 2 October 1997.

[6] A. Glavieux and C. Laot and J. Labat, "Turbo Equalization over a Frequency Selective Channel" *International Symposium on Turbo Codes, Brest, France*, pp. 96–102, 1997.

[7] M. S. Yee and L. Hanzo, "Multi-level radial basis function network based equalisers for Rayleigh channels," *Proc. of VTC'99, Houston, USA*, pp. 707–711, 16-19 May 1999.

[8] S. Chen, B. Mulgrew, and S. McLaughlin, "Adaptive Bayesian equalizer with decision feedback," *Trans. on Signal Processing*, vol. 41, no. 4, pp. 2918–2927, September 1993.

[9] E.-S. Chng and H. Yang and W. Skarbek, "Reduced complexity implementation of Bayesian equaliser using local RBF network for channel equalisation problem" *Electronics Letters*, vol 32, no 1, pp. 17-19, January 1996.

[10] S. K. Patra and B. Mulgrew, "Computational aspects of adaptive radial basis function equalizer design" *IEEE International Symposium on Circuits and Systems, ISCAS'97, Hong Kong*, vol 1, pp. 521-524, June 1997.

[11] P. Robertson, E. Villebrun and P. Hoeher, "A Comparison of Optimal and Sub-Optimal MAP Decoding Algorithms Operation in the Log Domain" *Proc. of the International Conference on Communications, Seattle, USA*, vol 2, pp. 1009-1013, June 1995.

[12] M. S. Yee and T. H. Liew and L. Hanzo, "Block Turbo Coded Burst-By-Burst Adaptive Radial Basis Function Decision Feedback Equaliser Assisted Modem" in*Proc. of VTC'99-Fall, Amsterdam, Netherlands*, vol 3, pp. 1600–1604, 19-22 September 1999.

[13] A. Klein, R. Pirhonen, J. Sköld and R. Suoranta, "FRAMES Multiple Access Model - Wideband TDMA with and without spreading," in *Proc. of PIMRC'97, Helsinki, Finland*, pp. 37–41, September 1997.