Correcting systematic mismatches in computed log-likelihood ratios [1]

AUGUSTUS J.E.M. JANSSEN, MARTEN VAN DIJK
Philips Research Laboratories, Prof. Holstlaan 4, 5656 AA Eindhoven
{*A.J.E.M.Janssen,Marten.van.Dijk*}*@philips.com*

ARIE KOPPELAAR
Philips Semiconductors System Laboratories, Prof. Holstlaan 4, 5656 AA Eindhoven
Arie.Koppelaar@philips.com

December 16, 2002

Abstract. A log-likelihood ratio (LLR) measures the reliability (and uncertainty) of a binary random variable being a zero versus being a one. LLRs are used as input in many implementations of decoding algorithms which also output LLRs. Mismatches in the outputs are for example generated by a decoder which is implemented by using approximations during its computations e.g. the symbol-by-symbol max-log a-posteriori probability (APP) algorithm versus the correct forward-backward (log-APP) algorithm or Hagenauer's approximation of the box function. We propose post-processing of output LLRs to correct part of the mismatches. This post-processing is a function of the statistics of the input LLRs. As examples, we study the effect of incorrectly scaled inputs to the box function leading to mismatched outputs, Hagenauer's approximation to the box function, and the effect of compensating mismatches of LLRs on the performance of iterative decoders.

1 INTRODUCTION

The log-likelihood ratio (LLR) of a binary random variable has found widespread usage in the decoding of binary codes [1]. A typical set-up is depicted in Figure 1.

Binary values $b_1, b_2, \ldots$ are transmitted over a channel and samples $z_1, z_2, \ldots$ are received. Symbols B_i and R_i denote the random variables corresponding to bit b_i and received value z_i. For each i, the LLR of B_i, conditioned on the knowledge of the received value R_i is computed[2]:

$$l_i = \log \frac{P(B_i = 0 | R_i = z_i)}{P(B_i = 1 | R_i = z_i)}.$$

The numbers $l_1, l_2, \ldots$ are fed to a decoder that computes LLRs of (linear combinations of) the transmitted bits.

One of the aims of this paper is to study the distortion of the output of the decoder if the inputs $l_1, l_2, \ldots$ are inaccurate. This situation can occur if the channel statistics are not precisely known. Another aim is to study the distortion in the decoder's output for several popular low-complexity approximations to genuine LLR computations such as the symbol-by-symbol max-log a-posteriori probability (max-log-APP) algorithm versus the correct forward-backward (log-APP) algorithm or Hagenauer's approximation of the box function.

In Section 2 we define LLRs and we expand on the channel model given in Figure 1. In Section 3 we give a general framework for studying the mismatches in output LLRs. We define systematic mismatches, we prove that a certain post-processing corrects this type of mismatch, and that other kinds of mismatches can not be corrected. This post-processing is used in Section 4 to analyse the distortion of the output of the box function in two variables if the two inputs are incorrectly scaled. In Section 5 we analyse Hagenauer's approximation of the box function and in Section 6 we discuss a simple network of Hagenauer's box functions. With simulations we show in Section 7 the performance gain that can be achieved in iterative decoders by compensating mismatched LLRs. We conclude in Section 8.

[2] In this paper log denotes the natural logarithm.

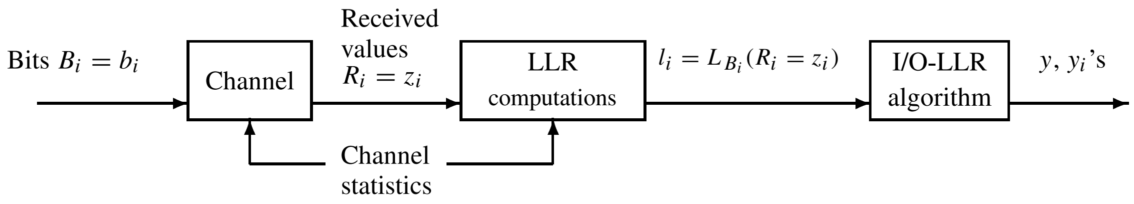

Figure 1: A data path.

2 LOG-LIKELIHOOD RATIOS

We introduce log-likelihood ratios (LLRs) and their properties by means of definitions and examples. This leads to the problem of analysing mismatches in LLRs discussed in this paper.

The LLR [2] of a binary random variable B conditioned on the knowledge of an instance of some other correlated random variable $R = z$ is defined by

$$L_B(R = z) \quad \doteq \quad \log \frac{P(B = 0|R = z)}{P(B = 1|R = z)}.$$

It represents soft information about B provided by our knowledge of $R = z$ and the statistics that relate B and R. If the conditional transition probability densities[3]

$$p(R = z|B = b)$$

and the source probabilities

$$P(B = b)$$

are known, then by Bayes' rule the LLR of B conditioned on $R = z$ can be computed as

$$L_B(R = z) = \log \frac{p(R = z|B = 0)}{p(R = z|B = 1)} + \log \frac{P(B = 0)}{P(B = 1)}. \quad (1)$$

2.1 LLRS ARE SUFFICIENT STATISTICS

We have seen how probabilities $P(B = 1|R = z)$ and $P(B = 0|R = z)$ define $L_B(R = z)$. An important property of LLRs is that $L_B(R = z)$ also defines the probabilities $P(B = 1|R = z)$ and $P(B = 0|R = z)$. By definition, $P(B = 1|R = z) \cdot e^{L_B(R=z)} = P(B = 0|R = z)$. From the definition of probabilities we infer $P(B = 0|R = z) + P(B = 1|R = z) = 1$. By combining both relations we obtain

$$P(B = b|R = z) = \frac{1}{1 + e^{\mu(b) \cdot L_B(R=z)}}, \qquad (2)$$

where $\mu(0) = -1$ and $\mu(1) = +1$ [2]. This shows how to compute the probability distribution $P(B = b|R = z)$ from LLRs. In this sense LLRs are sufficient statistics. We do not lose information by computing the LLR and discarding the received value.

[3]Function p denotes the probability density and function P denotes the probability.

2.2 THE BOX FUNCTION

In the next example we introduce the box function. Here the box function is used to compute, given input LLRs, a new LLR. The input LLRs are based on received values. It appears that the new LLR about a joint statistic of the received values can be computed by the box function transforming the input LLRs. Thus, the input LLRs are sufficient statistics representing the received values.

Suppose that $\{B_i\}_{i=1}^n$ is a sequence of binary uniformly distributed random variables representing a sequence of transmitted bits. Let $R = \{R_i\}_{i=1}^n$ be the sequence of random variables representing the received values. Hence, R_i and B_i depend statistically on one another, but they are statistically independent of any of the other random variables B_j and R_j, for $1 \leq j \neq i \leq n$. In this example we are interested in the binary random value $B = \sum\!\!\!\!\bigoplus_{i=1}^n B_i$ conditioned on the knowledge of the received sequence $R = z = \{z_i\}_{i=1}^n$. We want to compute the LLR

$$y = L_B(R = z) \quad = \quad L_{(\sum\!\!\!\!\bigoplus_{i=1}^n B_i)}(\{R_i = z_i\}_{i=1}^n).$$

By using (2), it can be shown that

$$L_{(\sum\!\!\!\!\bigoplus_{i=1}^n B_i)}(\{R_i = z_i\}_{i=1}^n) = F(l_1, \ldots, l_n),$$

where

$$F(l_1, \ldots, l_n) \doteq \log \frac{\sum_{(b_1,\ldots,b_n):\sum\!\!\!\!\bigoplus_{i=1}^n b_i = 0} \prod_{1 \leq j \leq n} e^{-b_j l_j}}{\sum_{(b_1,\ldots,b_n):\sum\!\!\!\!\bigoplus_{i=1}^n b_i = 1} \prod_{1 \leq j \leq n} e^{-b_j l_j}} \qquad (3)$$

with $l_i = L_{B_i}(R_i = z_i)$, $1 \leq i \leq n$. Function $F(.)$ is independent of the channel statistics. The channel statistics are contained in the LLRs l_i (they are sufficient statistics). Function $F(.)$ is called the box function and is introduced by Hagenauer et al. [2] in some elementary LLR calculations.

2.3 A LOG-APP DECODER

The next example discusses a log-APP decoder which as in Section 2.2 computes a new LLR based on input LLRs.

Let $\mathcal{C}$ be a binary linear code with code word length n. Suppose that with uniform probability a code word in $\mathcal{C}$ is selected and transmitted. Let the sequence $B = \{B_i\}_{i=1}^n$ correspond to the transmitted code word and let

$R = \{R_i\}_{i=1}^n$ correspond to the sequence of received values. We are interested in each of the binary random values B_i conditioned on the knowledge of the received sequence $R = z = \{z_i\}_{i=1}^n$ and conditioned on the knowledge $B = \{B_i\}_{i=1}^n \in \mathcal{C}$. We want to compute $L_{B_i}(R = z, B \in \mathcal{C})$, for $1 \le i \le n$.

A symbol-by-symbol a-posteriori probability (APP) decoder is an optimal soft-input soft-output decoder [3]. Such a decoder can work both in the probability domain (APP decoder) and in the LLR domain (log-APP decoder). Working in the LLR domain, the soft inputs and soft outputs of an optimal decoder are expected to represent LLRs. The log-APP decoder computes given the inputs

$$l_i = L_{B_i}(R_i = z_i), \quad 1 \le i \le n,$$

the outputs

$$\begin{aligned} y_i &= L_{B_i}(R = z, B \in \mathcal{C}), \quad 1 \le i \le n, \\ &= \log \frac{P(B_i = 0 | R = z, B \in \mathcal{C})}{P(B_i = 1 | R = z, B \in \mathcal{C})} \\ &= \log \frac{\sum_{(b_1,\ldots,b_n)\in\mathcal{C}:b_i=0} \prod_{1\le j\le n} e^{-b_j l_j}}{\sum_{(b_1,\ldots,b_n)\in\mathcal{C}:b_i=1} \prod_{1\le j\le n} e^{-b_j l_j}}, \end{aligned}$$

where the last equation follows from (2).

2.4 A complete data path

In Figure 1 a complete data path is depicted. We start transmitting a sequence of bits over a noisy channel, for example AWGN with BPSK transmission. The received values are transformed into LLRs. As seen from (1), the channel statistics are needed in these computations. A decoder uses the computed LLRs as inputs to compute LLRs of (linear combinations of) the transmitted bits. Such a decoder may be a network of box functions, see Section 2.2, or a log-APP decoder, see Section 2.3.

2.5 Mismatches

Consider an AWGN channel with BPSK transmission. Suppose that we use an incorrect estimate of the amplitude and/or variance with which we compute mismatched (incorrectly scaled) LLRs l_i's. Then the box function $F(l_1, \ldots, l_n)$ outputs a mismatched LLR. It will not compute the desired $y = L_B(R = z)$ but it computes

$$L_B(R = z) + \varepsilon(z),$$

where $\varepsilon(.)$ is the function representing the mismatch in the output. This situation for the box function in two variables is analysed in Section 4.

Mismatched inputs lead to mismatches in the output. Correct inputs but incorrect computations lead to mismatches in the output as well. For example, Hagenauer approximated the box function by replacing the sum operation in (3) by the max operation [2]. This results in the so called Hagenauer's box function. If we use the resulting function (which is easier to implement) we do not compute the desired $y = L_B(R = z)$ but we compute $L_B(R = z) + \varepsilon(z)$, where $\varepsilon(.)$ is the function representing the mismatch in the output in this situation. In Section 5 the mismatch due to use of Hagenauer's box function is analysed.

The forward backward algorithm [3] is an efficient implementation of the optimal log-APP decoding algorithm of convolutional codes. It can be approximated by using the max operation over exponentials, instead of summing exponentials. This leads to the so called max-log-APP algorithm or the modified soft-output Viterbi algorithm (SOVA) [1]. The used approximation introduces mismatches in the outputs.

In the iterative decoding of parallel concatenated codes, see [4, 5], the outputs of a log-APP decoder are used as inputs to a second log-APP decoder. If we use the max-log-APP algorithm instead, then the second decoder does not only use approximations leading to mismatches in the output LLRs, but also mismatched inputs leading to mismatches in the LLRs (in implementations we need finite precision which lead to approximations as well). Similarly, in a network of Hagenauer's box functions approximations and mismatched inputs lead to mismatched outputs. In Section 6 a simple network is considered and in Section 7 it is shown that scaling mismatched LLRs in iterative decoders increase the performance and that the scaling is related to the correction of mismatched LLRs.

3 Systematic mismatches

Let us consider the general situation in which we want to compute $L_B(R = z)$, but we actually compute

$$a = L_B(R = z) + \varepsilon(z),$$

we discard the received value z and keep the computed value a. Value $\varepsilon(z)$ represents the mismatch. Since we discard z, our remaining knowledge about z is that it is an element in $\mathcal{R}_a$, where

$$\mathcal{R}_a = \{\zeta : a = L_B(R = \zeta) + \varepsilon(\zeta)\}.$$

Define $q(\zeta) = p(R = \zeta)/p(R \in \mathcal{R}_a)$ for $\zeta \in \mathcal{R}_a$. Let

$$E(a) = \sum_{\zeta \in \mathcal{R}_a} q(\zeta)\varepsilon(\zeta),$$

and

$$\sigma(a) = \sqrt{\sum_{\zeta \in \mathcal{R}_a} q(\zeta)(\varepsilon(\zeta) - E(a))^2}.$$

$E(a)$ is the average mismatch over $\zeta \in \mathcal{R}_a$, and $\sigma(a)$ is the standard deviation of the considered mismatches. The value a does not contain any mismatch if both $E(a) = 0$ and $\sigma(a) = 0$. We say it contains a systematic mismatch if $E(a) \neq 0$. Similarly, it contains random mismatches if $\sigma(a) \neq 0$. In this concept $E(a)$ measures the extent to which value a contains a systematic mismatch, and $\sigma(a)$ measures the extent to which value a contains a random mismatch.

In Appendix A we prove the following theorem.

Theorem 1 *Let A be the random variable which corresponds to $a = L_B(R = z) + \varepsilon(z)$, and let*[4] $L(a) = \sum_{\zeta \in \mathcal{R}_a} q(\zeta) L_B(R = \zeta)$. *Then,*

$$|L_B(A = a) - L(a)| \leq \max_{\zeta \in \mathcal{R}_a} |\varepsilon(\zeta) - E(a)|,$$

and the first few terms of a Taylor series expansion of $L_B(A = a)$ is given by

$$L_B(A = a) = L(a) - \frac{1}{2} \frac{e^{L(a)} - 1}{e^{L(a)} + 1} \sigma(a)^2 + \ldots.$$

Suppose that the random mismatches are small, in the sense that[5]

$$\max_{\zeta \in \mathcal{R}_a} |\varepsilon(\zeta) - E(a)| \leq \varepsilon.$$

We derive (Theorem 1 is used in the second inequality), for each $z \in \mathcal{R}_a$,

$$\begin{aligned} & |L_B(A = a) - L_B(R = z)| \\ \leq \; & |L_B(A = a) - L(a)| + |L(a) - L_B(R = z)| \\ = \; & |L_B(A = a) - L(a)| + |\varepsilon(z) - E(a)| \\ \leq \; & 2 \max_{\zeta \in \mathcal{R}_a} |\varepsilon(\zeta) - E(a)| \leq 2\varepsilon, \end{aligned}$$

and

$$\begin{aligned} |a - L_B(R = z)| &= |\varepsilon(z)| \geq |E(a)| - |\varepsilon(z) - E(a)| \\ &\geq |E(a)| - \max_{\zeta \in \mathcal{R}_a} |\varepsilon(\zeta) - E(a)| \\ &\geq |E(a)| - \varepsilon. \end{aligned}$$

So, if $|E(a)| > 3\varepsilon$ then by computing $L_B(A = a)$ we correct some of the unwanted systematic mismatches. The Taylor series expansion of $L_B(A = a)$ in Theorem 1 seems to indicate this as well. The computation of $L_B(A = a)$ is a one-input one-output post-processing step in which the real valued output a is transformed into an other real value, a LLR.

If there are no mismatches at all we compute $L_B(A = a)$, where A is the random variable corresponding to $a = L_B(R = z)$. From the derivations in Appendix A it follows that $L_B(A = a) = a$. In other words, the post-processing function is equal to the identity function.

From Equation (2) we deduce that by computing $L_B(A = a)$ and discarding a we do not lose our knowledge about the statistics $P(B = b|A = a)$. From an information-theoretical point of view we do not lose any information; the uncertainty of B given our knowledge of a is equal to the uncertainty of B given our knowledge of $L_B(A = a)$. We conclude that systematic mismatches do not introduce uncertainty since we are able to correct them partly. This is in accordance with information theory: systematic errors simply do not introduce uncertainty while random errors do introduce uncertainty.

A natural choice for a post-processing function $f(.)$ minimises the average mismatch over $\zeta \in \mathcal{R}_a$, that is, it minimises

$$\sum_{\zeta \in \mathcal{R}_a} q(\zeta) |f(a) - L_B(R = \zeta)|.$$

Even though it minimises the average mismatch and reduces the systematic mismatch, we propose to use the post-processing $L_B(A = a)$ instead for the following reason. Let us consider the log-APP decoder of Section 2.3. If we use the LLRs $L_{B_i}(A_i = a_i)$ as inputs then the log-APP decoder computes LLRs $y_i = L_{B_i}(\{A_j = a_j\}_{j=1}^n, B \in \mathcal{C})$. In other words, conditioned on $B \in \mathcal{C}$ and given the knowledge of values $\{a_j\}_{j=1}^n$ we compute the probability $1/(1 + e^{-y_i})$ that bit $B_i = 0$ (see (2)). This is the most we can hope for. Our computations are optimal in this sense. In Appendix B we consider an example which demonstrates this advantage of using $L_B(A = a)$ instead of $f(a)$.

The proposed post-processing requires the precise knowledge of the channel statistics. In practise we do not have this knowledge, because otherwise we would avoid mismatches in the inputs and we would use better approximations. However, a proper analysis of the proposed post-processing does tell us the behaviour of systematic mismatches in the output (hopefully leading to improved implementations; see the next paragraph). To obtain some intuition about this behaviour we shall compute the post-processing function for the box function with incorrectly scaled inputs in Section 4 and we compute the post-processing function for Hagenauer's box function with correct inputs in Section 5. (In both cases we found exact expressions for very small a and for very large a.)

By means of simulations (of the whole data path from random variable B to random variable A; see Figure 1 where A corresponds to the output LLRs which contain mismatches) we estimate the probability density functions $p(A = a|B = 0)$ and $p(A = a|B = 1)$ with which we can determine $L_B(A = a)$ as a function of a. Note that it is only feasible to obtain $p(A = a|B = 0)$ and $p(A = a|B = 1)$ from simulations if these values are not too small. In practical implementations of decoding algorithms the output LLRs are quantised and therefore have a finite number of possible values. So post-processing can be implemented by means of a look-up table. This look-up table depends on the channel statistics. In practise, we are usually interested in a certain critical range of statistics for which the decoding should perform well. As far

[4] If necessary we may replace the sum in the definitions of $L(a)$, $E(a)$, and $\sigma(a)$ by integrals.

[5] Note that $\sigma(a) \leq \sum_{\zeta \in \mathcal{R}_a} q(\zeta) |\varepsilon(\zeta) - E(a)| \leq \max_{\zeta \in \mathcal{R}_a} |\varepsilon(\zeta) - E(a)|$.

as statistics outside this range are concerned, it will either be obvious that the performance is excellent or we do not mind if the performance is poor. So, the look-up table to be implemented must be adjusted to the critical range of statistics.

4 THE BOX FUNCTION WITH INCORRECTLY SCALED INPUTS

In this section we analyse the box function in two variables, which can be written as (see Section 2.2)

$$F(l_1, l_2) = \log \frac{1 + e^{l_1 + l_2}}{e^{l_1} + e^{l_2}}.$$

We assume an AWGN channel with BPSK transmission leading to

$$p(R_i = z | B_i = 0) = \frac{e^{-(z-\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}}$$

and

$$p(R_i = z | B_i = 1) = \frac{e^{-(z+\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}}.$$

Full knowledge of μ and σ^2 lead to full knowledge of $2\mu/\sigma^2$, which is necessary for transforming the received values into LLRs

$$L_{B_i}(R_i = z) = \frac{2\mu}{\sigma^2} z.$$

Having computed $l_1 = L_{B_1}(R_1 = z_1)$ and $l_2 = L_{B_2}(R_2 = z_2)$ we use them as arguments of the box function and compute $z = F(l_1, l_2) = L_{B_1 \oplus B_2}(R_1 = z_1, R_2 = z_2)$.

We now want to study what happens when the decoder has partial knowledge of the amplitude μ and/or variance σ^2 leading to a scaled transformation of the received values. Let α be the scaling factor leading to mismatched LLRs

$$\hat{L}_{B_i}(R_i = z) \doteq \frac{2\mu}{\alpha\sigma^2} z.$$

Let $\hat{L}_i$ represent the corresponding random variables. We use the mismatched LLRs as inputs to the box function. This leads to a systematic mismatch at the output of the box function. We want to compute the output $L_{B_1 \oplus B_2}(R_1 = z_1, R_2 = z_2)$. So the post-processing of the output suggested in the previous section is the function

$$L(z) \doteq L_{B_1 \oplus B_2}(Z = z) = \log \frac{p(Z = z | B_1 \oplus B_2 = 0)}{p(Z = z | B_1 \oplus B_2 = 1)},$$

where $Z = F(\hat{L}_1, \hat{L}_2)$.

In order to analyse the post-processing function we derive

$$\begin{aligned} p(\hat{L}_i = z | B_i = 0) &= \frac{\mathrm{d}}{\mathrm{d}z} P(\hat{L}_i \le z | B_i = 0) \\ &= \frac{\mathrm{d}}{\mathrm{d}z} P(R_i \le \frac{\alpha\sigma^2}{2\mu} z | B_i = 0) \\ &= \frac{\alpha\sigma^2}{2\mu} p(R_i = \frac{\alpha\sigma^2}{2\mu} z | B_i = 0) \\ &= \frac{e^{-(z-\beta)^2\alpha/4\beta}}{\sqrt{4\pi\beta/\alpha}}, \end{aligned}$$

where $\beta = 2\mu^2/(\alpha\sigma^2)$. We observe that $p(\hat{L}_i = z | B_i = 0) = p(\hat{L}_i = -z | B_i = 1)$ and

$$F(l_1, l_2) = -F(-l_1, l_2) = -F(l_1, -l_2).$$

From these symmetry considerations we deduce

$$L(z) = \log \frac{p(Z = z | B_1 = B_2 = 0)}{p(Z = -z | B_1 = B_2 = 0)}.$$

So, without loss of generality we assume that $B_1 = B_2 = 0$ and

$$Z = \log \frac{1 + e^{X+Y}}{e^X + e^Y},$$

where X and Y are independent Gaussian variables with mean β and variance $2\beta/\alpha$ such that

$$L(z) = \log \frac{p(Z = z)}{p(Z = -z)}.$$

The next theorem gives an expression of $p(Z = z)$; see Appendix C.1 for its proof.

Theorem 2 *The probability density function of Z can be determined as :*

$$p(Z = z) = e^{-\alpha\beta/2} \int_{-\infty}^{\infty} f_\alpha(e^t) p(W = t) p(W = t - z) dt,$$

where

$$p(W = w) = \begin{cases} 0, & w < 0 \\ \sqrt{\frac{2\alpha}{\pi\beta}} \cdot \frac{\exp\left(-\frac{\alpha}{2\beta}\log^2(e^w + \sqrt{e^{2w} - 1})\right)}{\sqrt{1 - e^{-2w}}}, & w > 0 \end{cases}$$

and

$$f_\alpha(x) = \cosh(\alpha \operatorname{arccosh}(x)), \quad x \ge 1.$$

Function $p(W = w)$ is the probability density function of $W = \log(\cosh(V))$ with V a Gaussian distributed random variable with mean 0 and variance β/α. It can be readily verified that:

$$p(Z = -z) = e^{-\frac{\alpha\beta}{2}} \int_{-\infty}^{\infty} f_\alpha(e^{t-z}) p(W = t) p(W = t - z) \mathrm{d}t.$$

Hence, when $\alpha = 1$ so that $f_\alpha(x) = x$, we obtain that $p(Z = z) = e^z p(Z = -z)$ and $L(z) = z$. If $\alpha = 1$ then the input LLRs do not contain mismatches, and therefore no post-processing is required as is expressed by the identity function $L(z) = z$.

It should be noted that $p(Z = z)$ has a logarithmic singularity at $z = 0$. Therefore, it is necessary to subject $p(Z = z)$ to a mathematical analysis to obtain more insight into the behaviour of $p(Z = z)$, especially at $z = 0$.

In Appendix C.2 and C.3 we show that for $\alpha \geq 1$ and $y \geq 1$,

$$y^\alpha f_\alpha(x) \leq f_\alpha(xy) \leq f_\alpha(x) f_\alpha(y), \quad x \geq 1, \qquad (4)$$

while for $\alpha < 1$ the inequality signs should be reversed. Applying this inequality with $f_\alpha(xy) = f_\alpha(e^t) = f_\alpha(e^{t-z}e^z)$ results for $\alpha \geq 1$ and $z > 0$ in the lower and upper bound

$$\alpha z \leq L(z) \leq \log f_\alpha(e^z). \qquad (5)$$

For $\alpha < 1$ the inequality signs should be reversed. Both the upper and the lower bound coincide for $\alpha = 1$ giving $L(z) = z$. Furthermore, it should be noted that the above inequality for $L(z)$ is sharp in the sense that $L(z) \to \alpha z$ when α is kept fixed and $\beta \to \infty$ while $L(z) \to \log f_\alpha(e^z)$ when α is kept fixed and $\beta \downarrow 0$. Experiments have shown that if β is not too small and α is fixed, the bound αz is close to being attained.

Both the upper bound for $\alpha \geq 1$ and the lower bound for $\alpha < 1$ of $L(z)$ exhibit linear behaviour for small z and for large z. This linear behaviour of these bounds can be made explicit by applying an upper bound on $f_\alpha(y)$ for $\alpha \geq 1$ and $y \geq 1$:

$$f_\alpha(y) \leq \begin{cases} y^{\alpha^2}, \\ 2^{\alpha-1} y^\alpha. \end{cases}$$

For $\alpha < 1$ the inequality signs should be reversed and we then obtain a lower bound on $f_\alpha(y)$. When we apply these bounds with $y = e^z$ and $\alpha \geq 1$ we obtain

$$L(z) \leq \log f_\alpha(e^z) \leq \begin{cases} \alpha^2 z, \\ \alpha z + (\alpha - 1)\log 2, \end{cases}$$

where the inequality signs should be reversed for $\alpha < 1$. In Appendix C.2 and C.3 we show that for fixed α it holds that

$$\lim_{z \to 0} \frac{L(z)}{z} = \alpha^2 \quad \text{and} \quad \lim_{z \to \infty} (L(z) - \alpha z) = (\alpha - 1)\log 2.$$

Hence, for large z the upper bound and the lower bound (5) on $L(z)$ have the same slope α and the bounds have a vertical offset of $(\alpha - 1)\log 2$.

The derived bounds of $L(z)$ are illustrated for $\alpha = 1/2$ in Figure 2. The case $\alpha > 1$ means that the inputs of the box function do not match $\mu^2/2\sigma^2$, the signal-to-noise-ratio (SNR) on the transmission channel, and that the true SNR is higher than assumed. This underestimation leads to outputs of the box function that are systematically too small (a factor of α smaller for large z and a factor of α^2 for small z). Similarly, for $\alpha < 1$ the outputs of the box function are systematically too large.

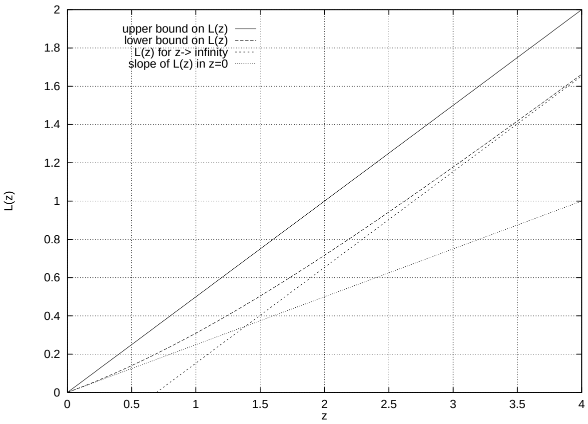

Figure 2: Bounds on the log-likelihood ratio of the box function with incorrectly scaled inputs.

In order to carry out the proposed post-processing we need full knowledge of α, in other words, we need to know $2\mu/\sigma^2$. If we would have known this ratio we would not have supplied incorrectly scaled inputs in the first place. The analysis shows how the linear behaviour of the systematic mismatch in the inputs propagates to the output. The outputs themselves will be used as inputs in future computations. The given analysis describes the systematic mismatches affecting these new inputs, and can help us understand the outputs of future computations.

5 Hagenauer's box function

In this section we shall analyse Hagenauer's approximation

$$H(l_1, \ldots, l_n) = \prod_{1 \leq i \leq n} (\text{sign}\, l_i) \cdot \min_{1 \leq j \leq n} |l_j|,$$

to the box function $F(l_1, \ldots, l_n)$ (See (3) in Section 2.2).

Let us consider a special case where $l_i = l \gg 0$ for all $1 \leq i \leq n$. Then, Hagenauer's approximation equals l. The true box function equals

$$\log \frac{\sum_{(b_1,\ldots,b_n):\bigoplus_{i=1}^n b_i = 0} \prod_{1 \leq j \leq n} e^{-b_j l}}{\sum_{(b_1,\ldots,b_n):\bigoplus_{i=1}^n b_i = 1} \prod_{1 \leq j \leq n} e^{-b_j l}}$$
$$\approx \quad \log \frac{1}{ne^{-l}} = l - \log n. \qquad (6)$$

Hagenauer's approximation thus produces a systematic mismatch $\approx -\log n$. That is, Hagenauer's approximation overestimates the output LLR (this corresponds to the results in [7]).

The general setting in this section is as in Section 2.2; $\{B_i\}_{i=1}^n$ is a sequence of binary uniformly distributed random variables representing a sequence of transmitted bits and $R = \{R_i\}_{i=1}^n$ represents the sequence of received values $z = \{z_i\}_{i=1}^n$. Random variables R_i and B_i depend statistically on one another, but they are statistically independ of any of the other random variables B_j and R_j, for $1 \leq j \neq i \leq n$. In the sequel L_i is the random variable corresponding to $L_{B_i}(R_i = z_i)$. Let H be the random variable corresponding to $H(L_1, \dots, L_n)$ and let $B = \sum\!\!\!\!\!\bigoplus_{i=1}^n B_i$. We compute $H(l_1, \dots, l_n)$ instead of $F(l_1, \dots, l_n) = L_B(R = z)$. So, the post-processing of the output suggested in Section 3 is the function $L_B(H = z)$. We redefine

$$L(z) \doteq L_B(H = z).$$

The next theorem shows that the proposed post-processing of Section 3 corrects the mismatch shown in (7). Its proof is given in Appendix D.

Theorem 3 *Suppose that all transition probability densities $p(R_i = r|B_i = b)$ are identical (independent of i). In addition we assume that these transition probability densities are symmetric so that*

$$p(R_i = r|B_i = 0) = p(R_i = -r|B_i = 1)$$

for all $1 \leq i \leq n$. Let

$$\mathcal{L} = \{\zeta : p(L_i = \zeta) \neq 0\}$$

(which is independent of i since all L_i's are identically distributed). Then, for $z \in \mathcal{L}$,

$$\begin{aligned} L(z) \quad &= \quad z - (\operatorname{sign} z) \log(1 + (n-1)\gamma(z)) \\ &- (\operatorname{sign} z) \left[(n-1)\gamma(z) + \frac{n(n-1)(n-2)\gamma(z)^3}{3 + 3(n-1)\gamma(z)} \right] e^{-2|z|} \\ &+ O(e^{-4|z|}), \end{aligned} \tag{7}$$

where

$$\gamma(z) = \frac{\int_{|z|}^{\infty} p(L_i = x|B_i = 0) e^{-x} \mathrm{d}x}{\int_{|z|}^{\infty} p(L_i = x|B_i = 0) e^{-|z|} \mathrm{d}x} \in (0, 1)$$

(which is independent of index i). Furthermore, $p(H = z|B = 0) = p(H = -z|B = 1)$, $L(-z) = -L(z)$, and $L(z)$ is a strictly increasing function in $z \in \mathcal{L}$. In particular,

$$\left. \frac{\mathrm{d}}{\mathrm{d}z} L(z) \right|_{z=0} = \left(\frac{1 - \gamma(0)}{1 + \gamma(0)} \right)^{n-1}.$$

Let us proceed with a specific choice for the probability densities $p(R_i = r|B_i = b)$ mentioned in Theorem 3. Take

$$p(R_i = z|B_i = 0) = \frac{e^{-(z-\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}}$$

and

$$p(R_i = z|B_i = 1) = \frac{e^{-(z+\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}},$$

representing BPSK transmission with additive white Gaussian noise. Then,

$$L_{B_i}(R_i = z) = \frac{2\mu}{\sigma^2} z.$$

Hence,

$$p(L_i = z|B_i = 0) = \frac{e^{-(z-\beta)^2/4\beta}}{\sqrt{4\pi\beta}},$$

where $\beta = 2\mu^2/\sigma^2$ (see also the previous section). We derive

$$\begin{aligned} \int_{|z|}^{\infty} e^{-x} p(L_i = x|B_i = 0) \mathrm{d}x &= \int_{-\infty}^{-|z|} p(L_i = x|B_i = 1) \mathrm{d}x \\ &= \int_{-\infty}^{-(|z|+\beta)/\sqrt{2\beta}} \frac{e^{-x^2/2}}{\sqrt{2\pi}} \mathrm{d}x, \\ &= \int_{(|z|+\beta)/\sqrt{2\beta}}^{\infty} \frac{e^{-x^2/2}}{\sqrt{2\pi}} \mathrm{d}x \end{aligned}$$

and

$$\int_{|z|}^{\infty} p(L_i = x|B_i = 0) \mathrm{d}x = \int_{(|z|-\beta)/\sqrt{2\beta}}^{\infty} \frac{e^{-x^2/2}}{\sqrt{2\pi}} \mathrm{d}x.$$

Let

$$Q(z) = \int_z^{\infty} \frac{e^{-x^2/2}}{\sqrt{2\pi}} \mathrm{d}x.$$

Then

$$\gamma(z) = \frac{Q((|z| + \beta)/\sqrt{2\beta})}{e^{-|z|} Q((|z| - \beta)/\sqrt{2\beta})}.$$

To obtain the asymptotic behaviour of $\gamma(z)$ we use the inequalities

$$(1 - 1/z^2) \frac{e^{-z^2/2}}{\sqrt{2\pi} z} < Q(z) < \frac{e^{-z^2/2}}{\sqrt{2\pi} z},$$

for $z > 0$ [6, pp. 31], leading to

$$\frac{|z| - \beta}{|z| + \beta} \cdot \frac{(|z| + \beta)^2 - 2\beta}{(|z| + \beta)^2} < \gamma(z)$$

and

$$\gamma(z) < \frac{|z| - \beta}{|z| + \beta} \cdot \frac{(|z| - \beta)^2}{(|z| - \beta)^2 - 2\beta},$$

for $|z| > \beta$. Hence (see also (7)),

$$\lim_{z \to \infty} (L(z) - z) = -\log n$$

and

$$\lim_{z \to -\infty} (L(z) - z) = \log n.$$

For small z other terms in the expansion of $L(z)$ become more important. Theorem 3 states

$$\lim_{z \to 0} \frac{L(z)}{z} = (1 - 2Q(\sqrt{\beta/2}))^{n-1}.$$

In Figure 3 we have plotted $L(z)$ for $n = 2$ and $\beta = 1$. A consequence of the proof in Appendix D is that $L(z) = z - \log(1+\gamma(z)) + \log(1+e^{-2z}\gamma(z))$ for $n = 2$ and $z \geq 0$, which is in agreement with (8). There is a vertical offset of $\log 2$ for large z, which can be corrected without any knowledge of the amplitude and variance. For decreasing z the offset decreases as a function of the SNR. If the SNR is known then we can implement a table which provides correction terms $L(z) - z$ for various z, so that we are able to reduce the systematic mismatch generated by Hagenauer's box function.

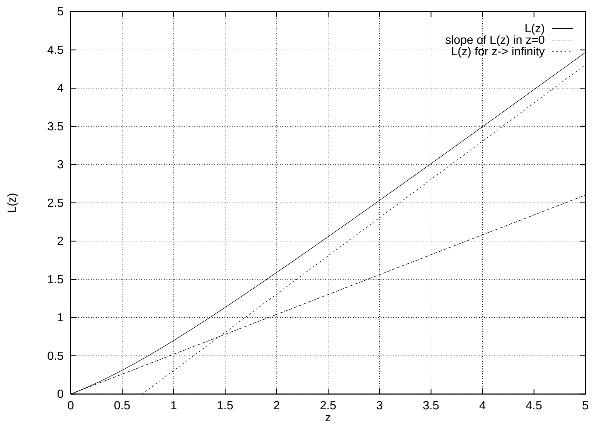

Figure 3: The log-likelihood ratio of Hagenauer's box function.

6 A NETWORK OF HAGENAUER'S BOX FUNCTIONS

From the definition of the box function $F(.)$ and from the definition of Hagenauer's box function $H(.)$ it follows that

$$F(l_1, l_2, l_3, l_4) = F(F(l_1, l_2), F(l_3, l_4))$$

and

$$H(l_1, l_2, l_3, l_4) = H(H(l_1, l_2), H(l_3, l_4)).$$

More generally, we can build a network using (Hagenauer's) box functions of two variables to implement a (Hagenauer's) box function in more than two variables. In this section we continue the discussion of Section 5 for a simple example of a network. We suppose that the assumptions in Theorem 3 hold.

Suppose that we compute $H(l_1, l_2, l_3, l_4)$ by computing $H(H(l_1, l_2), H(l_3, l_4))$. In Section 3 we propose to use the post-processing

$$L(z) = L_{B_1 \oplus B_2 \oplus B_3 \oplus B_4}(H(L_1, L_2, L_3, L_4) = z).$$

An other possibility is to compute $H(l_1, l_2)$ and $H(l_3, l_4)$ and not to postpone the post-processing but to apply the post-processing suggested in Section 3 on these values. This means that we twice use the post-processing function

$$\begin{aligned} \bar{L}(z) &= L_{B_1 \oplus B_2}(H(L_1, L_2) = z) \\ &= L_{B_3 \oplus B_4}(H(L_3, L_4) = z). \end{aligned}$$

This results in the post-processed values $\bar{L}(H(l_1, l_2))$ and $\bar{L}(H(l_3, l_4))$. Now we compute

$$H(\bar{L}(H(l_1, l_2)), \bar{L}(H(l_3, l_4)))$$

and we apply the final post-processing

$$\hat{L}(z) = L_{(B_1 \oplus B_2) \oplus (B_3 \oplus B_4)}(Z = z),$$

where Z is the random variable

$$Z = H(\bar{L}(H(L_1, L_2)), \bar{L}(H(L_3, L_4))).$$

We will prove that if we postpone all the post-processing we obtain the same final value, that is,

$$\begin{aligned} & L(H(H(l_1, l_2), H(l_3, l_4))) = \\ & \hat{L}(H(\bar{L}(H(l_1, l_2)), \bar{L}(H(l_3, l_4)))). \end{aligned}$$

According to Theorem 3 $\bar{L}(z)$ is a strictly increasing function in z. Hence, from the definition of Hagenauer's approximation to the box function it follows that

$$\begin{aligned} & H(\bar{L}(H(l_1, l_2)), \bar{L}(H(l_3, l_4))) = \\ & \quad \bar{L}(H(H(l_1, l_2), H(l_3, l_4))). \end{aligned} \tag{8}$$

A second consequence is

$$\begin{aligned} & L(H(H(l_1, l_2), H(l_3, l_4))) \\ = \; & L_{B_1 \oplus B_2 \oplus B_3 \oplus B_4}(H(H(L_1, L_2), H(L_3, L_4)) \\ & \qquad = H(H(l_1, l_2), H(l_3, l_4))) \\ = \; & L_{B_1 \oplus B_2 \oplus B_3 \oplus B_4}(\bar{L}(H(H(L_1, L_2), H(L_3, L_4))) \\ & \qquad = \bar{L}(H(H(l_1, l_2), H(l_3, l_4)))). \end{aligned}$$

By using (9) we derive

$$\begin{aligned} & L_{B_1 \oplus B_2 \oplus B_3 \oplus B_4}(\bar{L}(H(H(L_1, L_2), H(L_3, L_4))) \\ & \qquad = \bar{L}(H(H(l_1, l_2), H(l_3, l_4)))) \\ = \; & L_{(B_1 \oplus B_2) \oplus (B_3 \oplus B_4)}(H(\bar{L}(H(L_1, L_2)), \bar{L}(H(L_3, L_4))) \\ & \qquad = H(\bar{L}(H(l_1, l_2)), \bar{L}(H(l_3, l_4)))) \\ = \; & \hat{L}(H(\bar{L}(H_1(l_1, l_2)), \bar{L}(H_2(l_3, l_4))). \end{aligned}$$

This proves that in this specific example without loss of information we may postpone all the post-processing to the end. It remains an open problem whether this is true for more general networks of Hagenauer's box functions.

7 SIMULATION RESULTS

In this section we present simulation results that show the influence of compensating mismatched LLRs on the

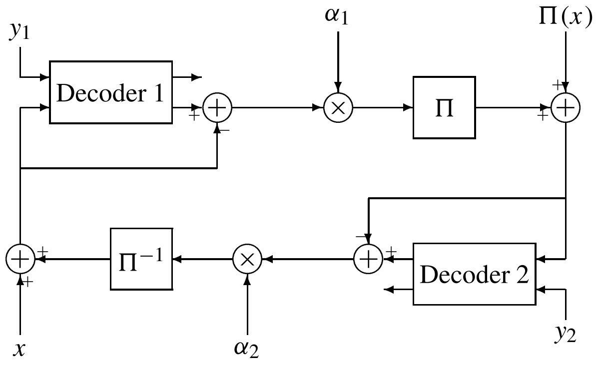

Figure 4: Schematic diagram of iterative decoder for PCCC

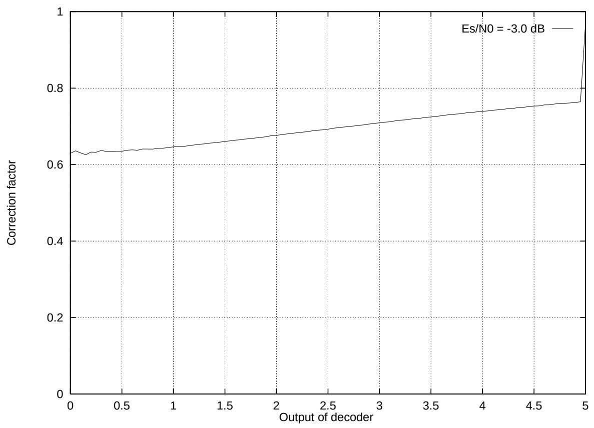

Figure 5: LLR correction of extrinsic information.

performance of an iterative decoder [2, 4, 9]. In Figure 4 a schematic diagram of an iterative decoder is shown. The symbols x, y_1 and y_2 represent the intrinsic information of respectively the information bits, the parity bits of encoder 1 and the parity bits of encoder 2. Compared to traditional iterative decoders, the iterative decoder shown in Figure 4 is extended with multipliers that scale the extrinsic information of the decoders 1 and 2 with α_1 and α_2, respectively. The use of scaling will be elucidated in the next section.

Reasons for arising mismatches of LLRs in iterative decoders maybe due to the fact that in the course of iterating the correlation between a-priori information and intrinsic information increases. Also the use of suboptimal component decoders like max-log-APP decoders will lead to mismatches. In particular the produced extrinsic information of max-log-APP decoders is an overestimate of the *true* LLR [7, 8]. The mismatch due to use of a max-log-APP decoder is quantified in the next section.

7.1 Quantifying mismatch of max-log-APP decoders

We quantify the mismatch of the extrinsic information produced by a max-log-APP decoder compared to the true LLR by determining the scaling factor that one has to apply in order to correct the relation between extrinsic information and LLR. For that purpose we do experiments with an 8-state recursive convolutional code with generator polynomial $G = (1, 15/13)$. By doing simulations we determined the distribution of the extrinsic information a produced by a max-log-APP decoder. This distribution is used to determine the LLR $L(a)$ of the extrinsic information a and can therefore be used for calculating the required correction factor $L(a)/a$ (see Section 3) of the extrinsic information. The required correction factor is shown in Figure 5 and turns out to be dependent on the decoder output a. The correction values are smaller than one and therefore validate the reported overestimation of LLRs in [7, 8]. An approximation to the non-constant scaling may be a constant correction factor.

7.2 Scaling in iterative decoders

In an iterative decoding scheme (see Figure 4), with component decoders of the max-log-APP type, we investigate experimentally whether compensation of mismatched LLRs lead to better performances. The used code is a parallel concatenated convolutional code with two 8-state recursive systematic component convolutional codes. The generator polynomials of the component codes are $G = (1, 15/13)$. The interleaver size is 150 and the two component codes are terminated separately by zero-tailing. The interleaver is a randomly chosen permutation. To avoid decoding complexity, the mismatched LLRs at the output of the component decoders are compensated with a constant scaling factor instead of using a scaling function like shown in Figure 5. For obtaining good scaling factors we use an experimental and greedy approach. We fix the SNR and start with one iteration and observe the performance (Bit Error Rate) at the output of decoder 2 while α_1 is varied. The scaling factor is varied in steps of 0.1 from 0.1 up to 1.0. The value of α_1 that gives the lowest BER is used in further simulations (but only for the first iteration). The same procedure is repeated for α_2 while we observe the BER at the output of decoder 1 and do 2 (actually 1.5) iterations. Then we vary α_1 again for the second iteration, etc. In this way we obtain two profiles of scaling factors for one SNR. For each SNR the procedure has to be repeated. The resulting scaling factors are shown in Table 1. We find that the scaling factor α_2 does not vary much with the SNR, but for α_1 we see that a higher SNR leads to higher values. Furthermore, the scaling factor found for α_1 in the first iteration matches well with the found scaling function shown in Figure 5.

7.3 Performance versus complexity

In Figure 6 the convergence behaviour of iterative decoding with scaling is shown. The figure gives the BER as function of the number of iterations. As a reference we also give the performances of iterative decoding without scaling

E_s/N_0	α_i	Iteration number									
		1	2	3	4	5	6	7	8	9	10
-4.0	1	0.7	0.7	0.7	0.7	0.7	0.6	0.6	0.7	0.6	0.7
	2	–	0.6	0.7	0.7	0.6	0.6	0.6	0.7	0.6	0.7
-3.0	1	0.8	0.7	0.8	0.8	0.8	0.8	0.8	0.7	0.8	0.7
	2	–	0.7	0.7	0.7	0.7	0.6	0.7	0.6	0.7	0.7
-2.0	1	0.8	0.9	0.9	1.0	1.0	–	–	–	–	–
	2	–	0.7	0.6	0.7	0.6	–	–	–	–	–

Table 1: Scaling factors for iterative decoding with max-log-APP

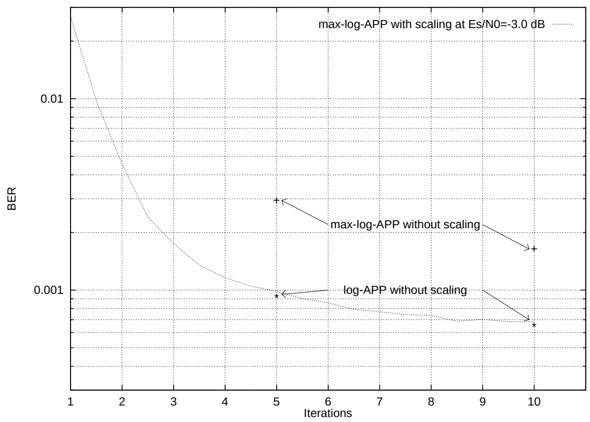

Figure 6: Convergence of iterative decoding with scaling at $E_s/N_0 = -3.0$ dB.

for 5 and 10 iterations. The results show that in case of iterative decoding with max-log-APP component decoders, due to properly scaling the required number of iterations can be halved for the same (or better) performance. Moreover, the performance approaches the performance of an iterative decoder with log-APP component decoders.

Notice that in [7] a different scaling is proposed that does not lead to significant performance improvements. The reason for that may be that they assume Gaussian distributed extrinsic information, while the non-constant correction factor shown in Figure 5 suggests a non-Gaussian distribution.

8 Conclusions

In general, systematic mismatches can be studied by means of a post-processing which corrects this type of mismatches. We demonstrated that this post-processing is generally not equivalent to minimising the average (systematic or non-systematic) mismatch.

For example, if we do not know the exact SNR we will incorrectly scale the inputs of the box function with say a factor α. This leads to a mismatched output of the box function. We have given upper and lower bounds on the output of the proposed post-processing as a function of the mismatch. If we estimate the SNR too small (underestimation) then the output of the box function is smaller than what it should be (a pessimistic output). Overestimation of the SNR leads to optimistic outputs of the box function. If the SNR is not too small and α is fixed, the post-processing is approximately equivalent to multiplying with the factor α. So the behaviour of the systematic mismatch at the inputs transfers to the output.

Hagenauer's approximation to the box function was considered as a second example. An expression of the post-processing function $L(z)$ has been derived. If z is not too small, this expression equals approximately $z - (\text{sign} z) \log n$, where n is the number of inputs. Hagenauer's approximation leads to a slightly pessimistic output. For a simple network of Hagenauer's box functions, we proved that without loss of information we may postpone all the post-processing to the end.

Simulation results show that our theory explains the influence of scaling on the performance of an iterative decoder. By using scaling factors the performance can be significantly improved.

Acknowledgements

We thank our colleague William Rey, whose simulations showed that the logarithmic singularity of $p(Z = z)$ at $z = 0$ may make the results of simulation unreliable, especially for z close to zero (see Section 4). We thank our colleague Ludo Tolhuizen for his suggestions which improved this paper.

A Properties of $L_B(A = a)$

In this appendix we present the proof of Theorem 1 concerning the post-processing $L_B(A = a)$, where A is the random variable corresponding to $a = L_B(R = z) + \varepsilon(z)$. We define $\mathcal{R}_a = \{\zeta : a = L_B(R = \zeta) + \varepsilon(\zeta)\}$ and $q(\zeta) = p(R = \zeta)/p(R \in \mathcal{R}_a)$ for $\zeta \in \mathcal{R}_a$.

We derive

$$L_B(A = a) \quad = \quad \log \frac{p(B = 0, R \in \mathcal{R}_a)}{p(B = 1, R \in \mathcal{R}_a)}$$

$$= \log \frac{\sum_{\zeta\in\mathcal{R}_a} q(\zeta)P(B=1|R=\zeta)}{\sum_{\zeta\in\mathcal{R}_a} q(\zeta)P(B=1|R=\zeta)}.$$

By using Equation (2) we obtain

$$\begin{aligned} L_B(A=a) &= \log \frac{\sum_{\zeta\in\mathcal{R}_a} q(\zeta)/(1+e^{-L_B(R=\zeta)})}{\sum_{\zeta\in\mathcal{R}_a} q(\zeta)/(1+e^{+L_B(R=\zeta)})} \\ &= \log \frac{\sum_{\zeta\in\mathcal{R}_a} q(\zeta)/(1+e^{-(a-\varepsilon(\zeta))})}{\sum_{\zeta\in\mathcal{R}_a} q(\zeta)/(1+e^{+(a-\varepsilon(\zeta))})}. \end{aligned}$$

Let $L(a) = \sum_{\zeta\in\mathcal{R}_a} q(\zeta)L_B(R=\zeta)$, and $E(a) = \sum_{\zeta\in\mathcal{R}_a} q(\zeta)\varepsilon(\zeta)$. We notice that $L_B(A=a) = \log\frac{1-x}{x}$, with $x = \sum_{\zeta\in\mathcal{R}_a} q(\zeta)/(1+e^{(a-\varepsilon(\zeta))})$, is a decreasing function in x. Hence, $a - \max_{\zeta\in\mathcal{R}_a}\varepsilon(\zeta) \le L_B(A=a) \le a - \min_{\zeta\in\mathcal{R}_a}\varepsilon(\zeta)$, and we obtain

$$-\max_{\zeta\in\mathcal{R}_a}(\varepsilon(\zeta)-E(a)) \le L_B(A=a) - L(a)$$

and

$$L_B(A=a) - L(a) \le -\min_{\zeta\in\mathcal{R}_a}(\varepsilon(\zeta)-E(a))$$

from which the first part of the theorem follows.

The second part of the theorem can be derived by taking partial derivatives of

$$x = \sum_{\zeta\in\mathcal{R}_a} q(\zeta)/(1+e^{(L(a)-(\varepsilon(\zeta)-E(a))}),$$

which is a function of variables $v_\zeta = \varepsilon(\zeta) - E(a)$, $\zeta\in\mathcal{R}_a$.

B $L_B(A=a)$ VERSUS $f(a)$

Assume an AWGN channel with BPSK transmission. Let $\mu = 1$ and $\sigma = 2.296069$ so that the SNR equals $\mu^2/2\sigma^2 = -10.23$ dB. Note that our example is not practical. If a zero is transmitted we receive an instance of a Gaussian distribution with mean μ and standard deviation σ. If a one is transmitted we receive an instance of a Gaussian distribution with mean $-\mu$ and standard deviation σ. Let B be the random variable corresponding to the transmitted bit.

Suppose that the received signal is quantised by 5 bits. That is, there are 32 quantisation intervals. For large enough SNR the performance of a convolutional code is given by the union bound. The intervals are chosen such that the union bound is minimised, or equivalently such that the minimum Bhattacharyya distance [6, pp. 311] between the signal sequences corresponding to two different code words is maximised. The quantisation intervals optimise the so-called cut-off rate [6, pp. 310–318]. Without much performance degradation we use the uniform spacing $\tau = 0.180\cdot\sigma$, that is the 32 quantisation intervals are $I_{-16} = (-\infty, -15\tau]$, $I_{-15} = (-15\tau, -14\tau]$, $I_{-14} = (-14\tau, -13\tau]$, …, $I_{-1} = (-\tau, 0]$, $I_1 = (0, \tau]$, …, $I_{16} = (15\tau, \infty)$. The quantised received signal z is represented by the index ($z \in \{-16, \ldots, -1, +1, \ldots, +16\}$) of the quantisation interval containing the originally received signal. Let R be the random variable corresponding to z. Let

$$L_z = \log\frac{P(B=0|R=z)}{P(B=1|R=z)}$$

be the LLR corresponding to the interval I_z and let $P_z = P(R=z)$ be the probability that we receive a signal in interval I_z; see Table 2. We notice that $L_{-z} = -L_z$ and $P_{-z} = P_z$.

z	L_z	P_z
1	0.07818243	0.06502740
2	0.2345484	0.06334471
3	0.3909203	0.06010470
4	0.5472841	0.05554249
5	0.7036525	0.04997747
6	0.8600187	0.04377633
7	1.016384	0.03731538
8	1.172751	0.03094384
9	1.329121	0.02495421
10	1.485495	0.01956311
11	1.641870	0.01490382
12	1.798244	0.01102980
13	1.954615	0.007926813
14	2.110979	0.005530324
15	2.267335	0.003744492
16	2.619154	0.006315138

Table 2: The statistics describing 32 quantisation intervals.

Suppose that we want to represent $R = z$ by means of 3 bits instead of 5 bits, which will lead to a mismatch. In this case we should have quantised the originally received signal by 3 bits right away. This would lead to a uniform spacing of $\tau = 0.569\cdot\sigma$, which is about 3 times the uniform spacing used for 32 quantisation intervals. Therefore, in order to represent $R = z$ by means of 3 bits we map $z \in J_1 = \{1, 2, 3\}$ into $a = 1$, we map $z \in J_2 = \{4, 5, 6\}$ into $a = 2$, we map $z \in J_3 = \{7, 8, 9\}$ into $a = 3$ and we map $z \in J_4 = \{10, \ldots, 16\}$ into $a = 4$. Similarly, $-z$ is mapped into $-a$. Let A be the random variable corresponding to a.

See Section 3, we propose the post-processing

$$\begin{aligned} L_B(A=a) &= \log\frac{P(A=a|B=0)}{P(A=a|B=1)} \\ &= \log\frac{\sum_{z\in J_a} P(R=z|B=0)}{\sum_{z\in J_a} P(R=z|B=1)} \\ &= \log\frac{\sum_{z\in J_a} P_z/(1+e^{-L_z})}{\sum_{z\in J_a} P_z/(1+e^{L_z})}, \end{aligned}$$

see (2). In this appendix we compare this to a different post-processing

$$f(a) = \sum_{z\in J_a} P_z L_z,$$

which minimises the average mismatch. Table 3 lists both function values.

a	$L_B(A=a)$	$f(a)$	$P(A=a\|B=0)$
-4	-1.780219	-1.819411	0.01991400
-3	-1.147889	-1.152016	0.04490553
-2	-0.6886431	-0.6913283	0.09982994
-1	-0.2295319	-0.2304663	0.1669406
+1	0.2295319	0.2304663	0.2100131
+2	0.6886431	0.6913283	0.1987626
+3	1.147889	1.152016	0.1415213
+4	1.780219	1.819411	0.1181130

Table 3: Two kinds of post-processing.

We make the example more explicit by supposing that we puncture the parities at the even positions of a rate 1/2 systematic non-recursive convolutional code with feedforward polynomial $1 + D$. The resulting punctured convolutional code words are a repetition of code words in the $[3, 2, 2]$ parity check code. In Section 2.3 we described a symbol-by-symbol log-APP decoder. Let $l_1, l_2, l_3 \in \{-4, -3, -2, -1, +1, +2, +3, +4\}$ be the three inputs. Then the first output equals $y_1 = l_1 + F(l_2, l_3)$. Based on y_1 we make a hard decision, if $y_1 < 0$ we decide a 1 was transmitted as first bit of the parity check code word and if $y_1 > 0$ we decide a 0 was transmitted as first bit of the parity check code word. Without loss of generality we assume the all-zero code word has been transmitted. Except for the 4 cases listed in Table 4 the two kinds of post-processing (PP) lead to the same hard decisions (HD's).

(l_1, l_2, l_3)	HD based on y_1 with the PP $L_B(A=a)$	HD based on y_1 with the PP $f(a)$
$(-3, -4, -4)$	1	0
$(-3, +4, +4)$	1	0
$(+3, -4, +4)$	0	1
$(+3, +4, -4)$	0	1

Table 4: Differences in the hard decisions (HD's).

The bit error rate (BER) after taking the hard decisions is equal to 0.3228271 in the case of the post-processing $L_B(A = a)$ and 0.3228486 in the case of the post-processing $f(a)$. Our proposed post-processing performs slightly better. This confirms the conclusion that the post-processing $L_B(A = a)$ is optimal. We notice that if we do not perform any decoding then the BER after taking hard decisions is equal to 0.3315900. So, our example is not practical but it does demonstrate the differences between the two post-processings.

C Analysis of random variable Z

In this appendix we present the proofs of the results given in Section 4 concerning the probability density function (pdf) $p(Z = z)$ and LLR $L(z) = \log\big(p(Z = z)/p(Z = -z)\big)$ of

$$Z = \log\left[\frac{1 + e^{X+Y}}{e^X + e^Y}\right], \tag{9}$$

where X, Y are independent Gaussian random variables with mean μ and variance σ^2. For this a careful analysis of the functions

$$f_\alpha(x) = \cosh(\alpha \operatorname{arccosh}(x)), \quad x \geq 1, \tag{10}$$

where $\alpha = 2\mu/\sigma^2$, is required. We note that μ and σ play a different role in Section 4. Yet, for a better understanding, we shall use the same variables in the context of this appendix. In Section 4, we apply the results of this appendix for $\mu = \beta$ and $\sigma^2 = 2\beta/\alpha$.

C.1 Explicit formula for $p(Z = z)$ and some first consequences

We shall show that for $z \in \mathbb{R}$

$$p(Z = z) = e^{-\frac{\mu^2}{\sigma^2}} \int_{-\infty}^{\infty} f_\alpha(e^t)\, p(W = t)\, p(W = t - z)\, \mathrm{d}t. \tag{11}$$

Here f_α is given by (11) and

$$p(W = w) = \begin{cases} 0, & w < 0, \\ \frac{2}{\sigma\sqrt{\pi}} \frac{\exp\left(-\frac{1}{\sigma^2}\log^2\left(e^w + \sqrt{e^{2w}-1}\right)\right)}{\sqrt{1-e^{-2w}}}, & w > 0, \end{cases} \tag{12}$$

is the pdf of $W = \log(\cosh(V))$ with V a Gaussian random variable with mean 0 and variance $\frac{1}{2}\sigma^2$.

Indeed, from (10) it follows that

$$\begin{aligned} Z &= \log\left[\frac{\cosh\frac{1}{2}(X+Y)}{\cosh\frac{1}{2}(X-Y)}\right] \\ &= \log(\cosh(U)) - \log(\cosh(V)), \end{aligned} \tag{13}$$

where $U = \frac{1}{2}(X + Y)$ and $V = \frac{1}{2}(X - Y)$ are independent Gaussian random variables with variance $\frac{1}{2}\sigma^2$ and with mean μ and 0, respectively. Write

$$T = \log(\cosh(U)), \quad W = \log(\cosh(V)).$$

For the pdf of W we compute

$$P(W \leq w) = \frac{2}{\sigma\sqrt{\pi}} \int_0^{v(w)} e^{-v^2/\sigma^2}\, \mathrm{d}v, \quad w > 0,$$

and $P(W \leq w) = 0$ for $w < 0$, where

$$v(w) = \operatorname{arccosh}(e^w) = \log\left(e^w + \sqrt{e^{2w} - 1}\right), \quad w > 0.$$

Thus

$$p(W=w)=\frac{2}{\sigma\sqrt{\pi}}e^{-v^2(w)/\sigma^2}v'(w),\quad w>0,$$

and it follows that $p(W=w)$ is given by (13).

For the pdf of T we compute

$$\begin{aligned} P(T\le t) &= \frac{1}{\sigma\sqrt{\pi}}\int_{-v(t)}^{v(t)} e^{-(u-\mu)^2/\sigma^2}\,\mathrm{d}u \\ &= \frac{2}{\sigma\sqrt{\pi}}e^{-\mu^2/\sigma^2}\int_0^{v(t)} e^{-u^2/\sigma^2}\cosh(\alpha u)\,\mathrm{d}u \end{aligned}$$

for $t>0$, and $P(T\le t)=0$ for $t<0$. From this we easily obtain

$$p(T=t)=e^{-\mu^2/\sigma^2}f_\alpha(e^t)p(W=t),\quad t>0,$$

with f_α as given in (11). Then, from (14) and the statistical independence of T and W, both

$$p(Z=z)=\int p(T=t)p(W=t-z)\,\mathrm{d}t,\quad z\in\mathbb{R},$$

and (12) follow.

We give some consequences of (12-13). By taking $t+z$ as integration variable in the integral in (12) with z replaced by $-z$ we obtain

$$p(Z=-z)=e^{-\frac{\mu^2}{\sigma^2}}\int_{-\infty}^{\infty}f_\alpha(e^{t-z})p(W=t)p(W=t-z)\,\mathrm{d}t.$$

Hence, we obtain for the LLR

$$\begin{aligned} L(z) &= \log\left[\frac{p(Z=z)}{p(Z=-z)}\right] \\ &= \log\left[\frac{\int f_\alpha(e^t)p(W=t)p(W=t-z)\,\mathrm{d}t}{\int f_\alpha(e^{t-z})p(W=t)p(W=t-z)\,\mathrm{d}t}\right], \\ & \quad z\ge 0. \end{aligned} \tag{14}$$

In particular, when $\alpha=1$, so that $f_\alpha(x)=x$, we find that $L(z)=z$.

We furthermore observe that, due to the factor $(1-\exp(-2w))^{-1/2}$ in the second member of the right-hand side of (13) and $f_\alpha(1)=1$,

$$\lim_{z\to 0}\frac{p(Z=z)}{\left|\log|z|\right|}=\frac{2}{\sigma^2\pi}e^{-\mu^2/\sigma^2}.$$

Hence $p(Z=z)$ has a logarithmic singularity at $z=0$.

C.2 Properties of f_α

To obtain inequalities for and insight into the behaviour of $L(z)$, we consider formula (15), in which we write

$$f_\alpha(e^t)=\frac{f_\alpha(e^{t-z}e^z)}{f_\alpha(e^{t-z})}f_\alpha(e^{t-z}),\quad t\ge z\ge 0.$$

We are therefore particularly interested in the lower and upper bounds of $f_\alpha(xy)/f_\alpha(x)$ as a function of $x=e^{t-z}\ge 1$ for a given value of $y=e^z\ge 1$. These bounds translate directly into bounds for $L(z)$ through formula (15).

Lemma 4

(i) *Let $y>1$ and $\alpha>1$. Then $f_\alpha(xy)/f_\alpha(x)$ decreases from $f_\alpha(y)$ to y^α when x increases from 1 to ∞.*

(ii) *Let $y>1$ and $\alpha<1$. Then $f_\alpha(xy)/f_\alpha(x)$ increases from $f_\alpha(y)$ to y^α when x increases from 1 to ∞.*

(iii) $f_1(x)=x$, $x\ge 1$; $f_\alpha(1)=1$, $f_\alpha'(1)=\alpha^2$, $\alpha>0$.

(iv) *For $\alpha>1$, $x>1$, $y>1$ we have*

$$y^\alpha<\frac{f_\alpha(xy)}{f_\alpha(x)}<f_\alpha(y)<\min\left(y^{\alpha^2},2^{\alpha-1}y^\alpha\right). \tag{15}$$

(v) *For $\alpha<1$, $x>1$, $y>1$ we have*

$$y^\alpha>\frac{f_\alpha(xy)}{f_\alpha(x)}>f_\alpha(y)>\max\left(y^{\alpha^2},2^{\alpha-1}y^\alpha\right). \tag{16}$$

(vi) *For $0<\alpha<\infty$ we have*

$$\frac{f_\alpha(xy)}{f_\alpha(x)}=y^{\alpha^2}\left(1+\mathcal{O}\left[(x-1)(y-1)+(y-1)^2\right]\right) \tag{17}$$

for $x\ge 1$, $y\ge 1$.

(vii) *For $0<\alpha<\infty$ we have*

$$\frac{f_\alpha(xy)}{f_\alpha(x)}=2^{\alpha-1}y^\alpha\left(1+\mathcal{O}\left[(x-1)+y^{-2\gamma}\right]\right) \tag{18}$$

for $x\ge 1$, $y\ge 1$, where $\gamma=\min(1,\alpha)$.

(viii) *$f_\alpha(x)$ is strictly convex in $x\ge 1$ when $\alpha>1$ and strictly concave in $x\ge 1$ when $0<\alpha<1$.*

(ix) $\log f_\alpha(\exp(z))$ *is strictly concave in $z\ge 0$ when $\alpha>1$ and strictly convex in $z\ge 0$ when $0<\alpha<1$.*

Proof. (i) We have

$$\frac{\mathrm{d}}{\mathrm{d}x}\left[\frac{f_\alpha(xy)}{f_\alpha(x)}\right]=\frac{1}{f_\alpha^2(x)}\left(yf_\alpha'(xy)f_\alpha(x)-f_\alpha(xy)f_\alpha'(x)\right)$$

for $x,y>1$. Since

$$f_\alpha'(x)=\frac{\alpha\sinh(\alpha\operatorname{arccosh}(x))}{\sqrt{x^2-1}},\quad x>1,$$

we obtain after some computations

$$yf_\alpha'(xy)-f_\alpha(xy)f_\alpha'(x)=\frac{\alpha}{x}f_\alpha(x)f_\alpha(xy)\big(\psi_\alpha(xy)-\psi_\alpha(x)\big)$$

for $x,y>1$, where

$$\psi_\alpha(x)=\frac{x\tanh(\alpha\operatorname{arccosh}(x))}{\sqrt{x^2-1}},\quad x>1. \tag{19}$$

Hence $f_\alpha(xy)/f_\alpha(x)$ decreases in $x>1$ if and only if $\psi_\alpha(x)$ decreases in $x>1$.

We now change variables in (20) according to $a = \operatorname{arccosh}(x)$, so that a increases from 0 to ∞ as x increases from 1 to ∞, and we shall show that

$$\psi_\alpha(\cosh(a)) = \frac{\tanh(\alpha a)}{\tanh(a)}$$

decreases in $a > 0$. We compute

$$\frac{\mathrm{d}}{\mathrm{d}a}\left[\frac{\tanh(\alpha a)}{\tanh(a)}\right] = \frac{\alpha\sinh(2a) - \sinh(2\alpha a)}{2\cosh(\alpha a)^2\sinh(a)^2}. \tag{20}$$

Since $\alpha > 1$ we have by strict convexity of sinh on $(0, \infty)$ that the right-hand side of (21) is negative for $a > 0$. Therefore, $\psi_\alpha(x)$ and $f_\alpha(xy)/f_\alpha(x)$ decrease in $x > 1$.

It is not hard to see that

$$\left.\frac{f_\alpha(xy)}{f_\alpha(x)}\right|_{x=1} = f_\alpha(y), \quad \lim_{x\to\infty}\frac{f_\alpha(xy)}{f_\alpha(y)} = y^\alpha,$$

and this completes the proof of (i).

(ii) Along similar lines as the proof of (i).

(iii) Trivial.

(iv) The first two inequalities in (iv) follow from (i), so we only need to show that

$$f_\alpha(x) < \min\left(y^{\alpha^2}, 2^{\alpha-1}y^\alpha\right), \quad y > 1. \tag{21}$$

We compute

$$\frac{\mathrm{d}}{\mathrm{d}y}\left[\frac{f_\alpha(y)}{y^\alpha}\right] = \tag{22}$$
$$\frac{\alpha f_\alpha(y)}{y^\alpha\sqrt{y^2-1}}\big[\tanh(\alpha\operatorname{arccosh}(y)) - \tanh(\operatorname{arccosh}(y))\big],$$

$$\frac{\mathrm{d}}{\mathrm{d}y}\left[\frac{f_\alpha(y)}{y^{\alpha^2}}\right] = \tag{23}$$
$$\frac{\alpha f_\alpha(y)}{y^{\alpha^2}\sqrt{y^2-1}}\big[\tanh(\alpha\operatorname{arccosh}(y)) - \alpha\tanh(\operatorname{arccosh}(y))\big].$$

From strict monotonicity and strict concavity of tanh on $(0, \infty)$ and the fact that $\alpha > 1$ it then follows that the right-hand sides of (24) and (25) are positive and negative, respectively. Then (22) follows from

$$f_\alpha(1) = 1, \quad \lim_{y\to\infty}\frac{f_\alpha(y)}{y^\alpha} = 2^{\alpha-1}.$$

(v) Along similar lines as the proof of (iv).

(vi) We first observe that $f_\alpha(x)$ is a smooth function of $x \geq 1$, and that

$$f_\alpha(1) = 1, \quad f'_\alpha(1) = \alpha^2.$$

By the Taylor expansion of $f_\alpha(xy)$ around $y = 1$ (with $x \geq 1$ fixed), we have for $y \geq 1$

$$\begin{aligned} f_\alpha(xy) &= f_\alpha(x) + x(y-1)f'_\alpha(x) + \mathcal{O}\left((y-1)^2\right) \\ &= f_\alpha(x)\left[1 + \frac{xf'_\alpha(x)}{f_\alpha(x)}(y-1) + \mathcal{O}\left((y-1)^2\right)\right] \\ &= f_\alpha(x)\Big[1 + \alpha^2(y-1) + \\ &\qquad \mathcal{O}\left[(x-1)(y-1) + (y-1)^2\right]\Big]. \end{aligned}$$

The result then easily follows from

$$y^{\alpha^2} = \exp(\alpha^2\log y) = 1 + \alpha^2(y-1) + \mathcal{O}\left((y-1)^2\right).$$

(vii) There holds

$$\begin{aligned} f_\alpha(x) &= \frac{1}{2}\left[\left(x + \sqrt{x^2-1}\right)^\alpha + \left(x + \sqrt{x^2-1}\right)^{-\alpha}\right] \\ &= 2^{\alpha-1}x^\alpha\left(1 + \mathcal{O}(x^{-2} + x^{-2\alpha})\right) \\ &= 2^{\alpha-1}x^\alpha\left(1 + \mathcal{O}(x^{-2\gamma})\right), \quad x \geq 1, \end{aligned}$$

where $\gamma = \min(1, \alpha)$. Hence, for $x, y \geq 1$,

$$\begin{aligned} f_\alpha(xy) &= 2^{\alpha-1}y^\alpha x^\alpha\left(1 + \mathcal{O}((xy)^{-2\gamma})\right) \\ &= 2^{\alpha-1}y^\alpha f_\alpha(x)\,(1 + \mathcal{O}(x-1))\left(1 + \mathcal{O}((xy)^{-2\gamma})\right), \end{aligned}$$

and from this the result easily follows.

(viii) We compute for $x > 1$

$$f''_\alpha(x) =$$
$$\frac{\alpha x f_\alpha(x)}{(x^2-1)^{\frac{3}{2}}}\big[\alpha\tanh(\operatorname{arccosh}(x)) - \tanh(\alpha\operatorname{arccosh}(x))\big],$$

and the claims easily follow from strict concavity of tanh on $[0, \infty)$.

(ix) Assume that $\alpha > 1$. From (i) we have that

$$y{>}1,\ 1{<}\,x_1{<}\,x_2 \ \Rightarrow\ f_\alpha(x_1y)f_\alpha(x_2) > f_\alpha(x_2y)f_\alpha(x_1).$$

Taking

$$x_1 = e^v,\ x_2 = e^{v+w},\ y = e^z$$

with $v, w, z > 0$ and letting $g(z) = \log f_\alpha(\exp(z))$, we see that

$$v, w, z > 0 \ \Rightarrow\ g(v{+}z){+}g(v{+}w) > g(v{+}w{+}z){+}g(v).$$

This is equivalent to convexity of g on $(0, \infty)$. The proof for the case that $\alpha < 1$ is similar.

QED

We conclude this subsection with some comments. In terms of the hypergeometric functions F, see [10, Ch. 15], in particular 15.1.17 on p. 556, we have for $x \geq 1$

$$f_\alpha(x) = \cosh(\alpha\operatorname{arccosh}(x)) = F\left(-\alpha, \alpha; \frac{1}{2}; \frac{1}{2}(1-x)\right).$$

When $\alpha = 0, 1, \ldots$, we get, see [10, 22.5.47 on p.779 and 22.3.6 on p.775],

$$f_\alpha(x) = \Gamma_\alpha(x) = \frac{1}{2}\alpha\sum_{m=0}^{\lfloor\frac{1}{2}\alpha\rfloor}(-1)^m\frac{(\alpha-m-1)!}{m!(\alpha-2m)!}(2x)^{\alpha-2m},$$

the Chebyshev polynomial of the first kind of degree α.

Furthermore, the functions $f_\alpha(x)$ map $[1, \infty)$ onto $[1, \infty)$ and they form a group in the sense that

$$f_{\alpha\beta}(x) = f_\alpha(f_\beta(x)), \quad \alpha, \beta > 0; x \geq 1.$$

Interestingly, the functions y^{α^2} and $2^{\alpha-1}y^\alpha$ that occur at the right-hand sides of (16-17), considered as mappings of $[0,\infty)$ onto $[0,\infty)$, form groups in the same sense as well. A similar comment applies to the functions y^α, see left-hand sides of (16-17).

C.3 Consequences for the log-likelihood ratio

The inequalities (4) and (5) in the main text follow straightforwardly from (15) and the Lemma in C.2. For instance, when $\alpha > 1$, the first inequality in (4) follows from the first inequality in (16) so that

$$f_\alpha(e^t) = f_\alpha(e^{t-z}e^z) > e^{\alpha z} f_\alpha(e^{t-z}), \quad t \geq z > 0.$$

We shall now sketch proofs of the fact that if $\alpha = 2\mu/\sigma^2 > 0$ and $z > 0$ are fixed we have

$$\lim_{\sigma\downarrow 0} L(z) = \log\left[f_\alpha(e^z)\right], \quad \lim_{\sigma\to\infty} L(z) = \alpha z. \tag{24}$$

We shall also sketch proofs of the fact that if $\alpha > 0$ and $\sigma > 0$ are fixed we have

$$\lim_{z\downarrow 0} \frac{L(z)}{z} = \alpha^2, \quad \lim_{z\to\infty}\big(L(z) - \alpha z\big) = (\alpha-1)\log 2. \tag{25}$$

The proofs of these limit relations consist of a consideration of the limiting behaviour of the pdf

$$\frac{p(W=t)p(W=t-z)}{\int_z^\infty p(W=t)p(W=t-z)\,\mathrm{d}t}, \quad t \geq z, \tag{26}$$

for each of the four cases in (25-26), together with an appeal to the appropriate items in the Lemma in C.2.

As for the first limit in (25), we note that for fixed $\alpha > 0$, $z > 0$ the pdf in (27) concentrates all its mass at the point $t = z$ when $\sigma \downarrow 0$, see (13). Hence, in the right-hand side of (15) we can replace the $f_\alpha(e^t)$ in the numerator by $f_\alpha(e^z)$ and the $f_\alpha(e^{t-z})$ in the denominator by $f_\alpha(1) = 1$.

Similarly, for the second limit in (25), we note that for fixed $\alpha > 0$, $z > 0$ and any arbitrarily large $T > 0$ the pdf in (27) concentrates all its mass in $[T,\infty)$ as $\sigma \to \infty$. Since by (i) and (ii) of the Lemma in C.2 we have

$$\lim_{x\to\infty} \frac{f_\alpha(xy)}{f_\alpha(x)} = y^\alpha,$$

we can replace the $f_\alpha(e^t)$ in the numerator of the right-hand side of (15) effectively by

$$f_\alpha(e^{t-z}e^z) = e^{\alpha z} f_\alpha(e^{t-z}),$$

and this yields the second limit formula in (25).

As for the first limit in (26), we observe that for fixed $\alpha > 0$, $\sigma > 0$ the pdf in (27) concentrates all its mass at $t = z$ when $z \downarrow 0$. This is because of the occurrence of the factor $(1-\exp(-2w))^{-1/2}$ in the lower line of the expression (13) for $p(W=w)$. Hence, by (18), we can replace the $f_\alpha(e^t)$ in the numerator of the right-hand side of (15) effectively by

$$f_\alpha(e^{t-z}e^z) = e^{\alpha^2 z} f_\alpha(e^{t-z}).$$

Finally, for the second limit in (26) we note that for fixed $\alpha > 0$, $z > 0$ the pdf in (27) concentrates all its mass at $t = z$ when $z \to \infty$. This is because of the occurrence of the $\log^2\big(\exp(w) + (\exp(2w)-1)^{1/2}\big)$ in the exponent in the lower line of the expression (13) for $p(W=w)$. Hence, by (19), we can replace the $f_\alpha(e^t)$ in the numerator of the right-hand side of (15) effectively by

$$f_\alpha(e^{t-z}e^z) = 2^{\alpha-1}e^{\alpha z}.$$

This completes the sketches of the proofs of all four limit relations in (25-26).

D Analysis of Hagenauer's box function

In this appendix we prove Theorem 3 in which we analyse $L(z) = L_B(H=z)$. By using the assumptions in Theorem 3, a straightforward case analysis proves

$$\begin{aligned} &P(H \geq z|B=0) \\ &= \begin{cases} \frac{(A(z)+B(z))^n+(B(z)-A(z))^n}{2}, & z \geq 0, \\ 1 - \frac{(A(-z)+B(-z))^n-(B(-z)-A(-z))^n}{2}, & z < 0, \end{cases} \end{aligned}$$

where $A(z) = P(L_i \geq z|B_i = 1) = P(L_i \leq -z|B_i = 0)$ and $B(z) = P(L_i \geq z|B_i = 0) = P(L_i \leq -z|B_i = 1)$. Hence,

$$\begin{aligned} & P(H = z|B=0) \\ = \; & \frac{\mathrm{d}}{\mathrm{d}z} P(H \geq z|B=0) \\ = \; & -\frac{n}{2}\{(A(|z|)+B(|z|))^{n-1}(A'(|z|)+B'(|z|)) \\ & -(\operatorname{sign} z)\cdot(B(|z|)-A(|z|))^{n-1}(B'(|z|)-A'(|z|))\}, \end{aligned}$$

which equals

$$\begin{gathered} \tfrac{n}{2}p(L_i = |z||B_i = 0)\cdot \\ \left(\left[\int_{|z|}^\infty (1+e^{-x})p(L_i = x|B_i=0)\mathrm{d}x\right]^{n-1}(1+e^{-|z|}) + \right. \\ \left. \operatorname{sign}(z)\left[\int_{|z|}^\infty (1-e^{-x})p(L_i = x|B_i=0)\mathrm{d}x\right]^{n-1}(1-e^{-|z|})\right). \end{gathered}$$

By using similar arguments we find that $p(H = z|B = 1) = p(H = -z|B = 0)$. Since $z \in \mathcal{L}$ if and only if $p(L_i = |z||B_i = 0) \neq 0$ we infer from the derived equations that $z \in \mathcal{L}$ if and only if $p(H = z|B = 0) \neq 0$ and if and only if $p(H = z|B = 1) \neq 0$. Hence, for $z \in \mathcal{L}$,

$$\begin{aligned} L(z) &= L_B(H=z) = \log\frac{D(|z|) + (\operatorname{sign} z)K(|z|)}{D(|z|) - (\operatorname{sign} z)K(|z|)} \\ &= (\operatorname{sign} z)\log\frac{D(|z|)+K(|z|)}{D(|z|)-K(|z|)}, \end{aligned}$$

where

$$D(z) = \frac{1}{2}\left[\frac{\int_{|z|}^{\infty}(1+e^{-x})p(L_i = x|B_i = 0)\mathrm{d}x}{\int_{|z|}^{\infty} p(L_i = x|B_i = 0)\mathrm{d}x}\right]^{n-1}(1+e^{-|z|})$$
$$= \frac{1}{2}(1 + e^{-|z|})\sum_{j=0}^{n-1}\binom{n-1}{j}\gamma(z)^j e^{-j|z|},$$

with $\gamma(z)$ being defined in Theorem 3, and

$$K(z) = \frac{1}{2}(1 - e^{-|z|})\sum_{j=0}^{n-1}\binom{n-1}{j}(-1)^j\gamma(z)^j e^{-j|z|}.$$

This results in the following expression,

$$L(z) = z - (\text{sign } z)\log\frac{\sum_{i=0}^{\lceil n/2\rceil}(a_{2i}(z) + a_{2i+1}(z))e^{-2i|z|}}{\sum_{i=0}^{\lceil n/2\rceil}(a_{2i-1}(z) + a_{2i}(z))e^{-2i|z|}},$$

where $a_{-1}(z) = a_n(z) = 0$ and

$$a_i(z) = \binom{n-1}{i}\gamma(z)^i,$$

for $0 \leq i \leq n-1$. By using this expression and by noting that $\log(1+x) = x + O(x^2)$, the first part of the theorem follows.

It remains to be proven that $L(z)$ is strictly increasing in $z \in \mathcal{L}$. This follows from the following observations. We note that for $0 < z \in \mathcal{L}$,

$$\frac{\mathrm{d}}{\mathrm{d}z}L(z) = 2\frac{D(z)K(z)}{D(z)^2 - K(z)^2}\cdot\frac{\mathrm{d}}{\mathrm{d}z}\log\frac{K(z)}{D(z)},$$

$D(z) > K(z) > 0$, and

$$\log\frac{K(z)}{D(z)} = (n-1)\log\frac{\int_{|z|}^{\infty}(1 - e^{-x})p(L_i = x|B_i = 0)\mathrm{d}x}{\int_{|z|}^{\infty}(1 + e^{-x})p(L_i = x|B_i = 0)\mathrm{d}x}$$
$$+ \log\frac{1 - e^{-|z|}}{1 + e^{-|z|}}$$

is strictly increasing in $0 < z \in \mathcal{L}$. A similar argument holds for $0 > z \in \mathcal{L}$. We note that the derivative of $L(z)$ in $z = 0$ equals ($K(0) = 0$)

$$\frac{2K'(0)}{D(0)} = \left(\frac{1-\gamma(0)}{1+\gamma(0)}\right)^{n-1}.$$

REFERENCES

[1] S. Lin, T. Kasami, T. Fujiwara, and M. Fossorier. *Trellis and Trellis-Based Decoding Algorithms for Linear Block Codes.* Kluwer Academic Publishers, 1998, pp. 247–252.

[2] J. Hagenauer, E. Offer, and L. Papke. Iterative decoding of binary block and convolutional codes. *IEEE Trans. on Information Theory*, 42:429–445, March 1996.

[3] L.R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Trans. on Information Theory*, 20:284–287, March 1974.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: Turbo codes. In *Proc. of IEEE Int. Conf. on Communications*, pages 1064–1070, Geneva, Switzerland, 1993.

[5] J. Hagenauer. The turbo principle: Tutorial introduction and state of the art. In *Proc. of Int. Symp. on Turbo Codes and Related Topics*, pages 1–11, Brest, France, 1997.

[6] S.G. Wilson. *Digital Modulation and Coding.* Prentice Hall, Inc., 1996.

[7] L. Papke, P. Robertson, and E. Villebrun. Improved decoding with the SOVA in a parallel concatenated (turbo-code) scheme. In *Proc. of IEEE Int. Conf. on Communications*, pages 102–106, 1996.

[8] L. Lin, and R.S. Cheng. Improvements In SOVA-Based Decoding For Turbo Codes. In *Proc. of IEEE Int. Conf. on Communications*, pages 1473–1478, Montreal, Canada 1997.

[9] S. Crozier, K. Gracie, and H. Hunt. Efficient Turbo Decoding Techniques. In *Proc. of Int. Conf. on Wireless Communciations*, Calgary, Canada, July 1999.

[10] M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions.* Dover, New York, 1970.