

Joint Carrier Phase Estimation and Turbo Decoding Using Bit-Carrier-Phase APP Decoder

Amir Saroka and Dan Raphaeli*, IEEE Senior Member

Abstract – In this paper we present an algorithm for joint carrier phase estimation and turbo decoding for the case of rapidly varying carrier phase during the transmitted block. The proposed algorithm shows improved performance over previously proposed communication schemes, both coherent and non-coherent, for channels with Additive White Gaussian Noise (AWGN) and high carrier phase noise. The novel algorithm utilizes a modified “two dimensional” Bit-Carrier-phase A Posteriori Probability (BCAPP) decoder containing additional states representing the received carrier phase. The BCAPP decoder calculates two extrinsic metrics; one representing the bit soft value and the other representing the received carrier phase Probability Density Function (PDF) approximation. A modified structure of the turbo code iterations is suggested, implementing separate propagation of the two metrics between the BCAPP decoders. Most known methods to combat very high phase noise in the channel at low signal to noise ratio are based on non-coherent demodulation. Such reception schemes are designed for the worst-case phase noise and suffer high degradation for any level of phase noise. In our scheme, together with improved performance in high phase noise, the degradation decreases with the actual channel phase noise level. We show the robustness against phase noise model mismatch.

I. INTRODUCTION

Turbo Codes can achieve low bit error rates at astonishingly low signal to noise ratio (SNR) but with great sensitivity to carrier phase errors [2]. Conventional coherent carrier phase synchronization is difficult or impractical at such low SNR especially when using high phase noise sources, forcing non-coherent reception in many cases. Bursty frequency hopping systems (using synthesized sources for wide frequency band) and high carrier frequency transmissions are examples of applications that often suffer from high phase noise. Another example is deep-space transmission where extremely low SNR (using efficient codes with low rates) is used, combined with low bit rate and high carrier frequency.

* Amir Saroka and Dr. Dan Raphaeli are with the School of Electrical Engineering, Faculty of Engineering, Tel-Aviv University, Tel-Aviv 69978, Israel, email: danr@eng.tau.ac.il

Recently, new improved iterative methods were suggested for carrier phase recovery. Some of the methods use an outer encoder (convolutional, LDPC or turbo code) followed by a modulation encoder, with iterative decoding between them on the receiver side [1], [5], [7], [8], [9], [12], [13]. Typically the outer and modulation encoders are separated by an interleaver. Most modulation encoders use differential encoding (DE) or orthogonal waveforms [13] that assist non-coherent detection. Such modulations tend to decrease the code performance relative to Binary Phase Shift Keying (BPSK) for the same bandwidth expansion.

The decoding of the inner code by the Modulation Decoder (MD) is designed for channel with phase noise. An example of such method is Multiple Symbol Differential Detection (MSDD), which is based on the assumption that carrier phase is constant over L symbols, where L varies between 3 to few tens of symbols [10], [11]. Another example is [1] wherein the differential decoder was implemented using the BCJR (Bahl, Cocke, Jelinek and Raviv) algorithm [6], on the phase trellis. These methods start the first decoding iteration with MD, which produces estimation without using the outer-code information. The noisy Log Likelihood Ratio (LLR) estimates at low SNR in the first iteration are too poor for the iterative algorithm to overcome and converge. Other methods jointly estimate the bit and phase simultaneously, avoiding the serial concatenation [3], [4]. These methods are more complex but provide improved performance. Most of them estimate the phase while deriving the extrinsic information of the bit using a modified Soft-In-Soft-Out (SISO) decoder. In [3], a phase search is used to maximize the bit metric in each modified SISO decoder independently, passing only bit metrics between decoders. In [4], partial information about the phase is passed from one SISO decoder to the other. This article presents an algorithm of the last kind and suggests a new method for passing the extrinsic information of the phase between the modified BCAPP decoders.

In this article we present the Joint Carrier Phase Estimation and Turbo Decoder (JCPETD) algorithm. The JCPETD algorithm is using the proposed Bit Carrier phase A Posteriori Probability (BCAPP) decoder designed to produce the two soft metrics, one for the carrier phase and one for the data bit value. The two metrics propagate separately from one BCAPP block to the other, while only the bit value metric is interleaved back and forth between the two BCAPP blocks. The new BCAPP decoder ensures that both parameters of carrier phase and turbo decoded bit are estimated using maximum information from each single component code at

every step of the decoding process, therefore resulting in better performance than previously proposed algorithms.

In many cases the phase noise is a priori unknown or varies in time or temperature. When phase noise is low, most non-coherent methods result in inferior performance. The JCPETD algorithm is relatively robust to unknown phase noise variance. Even if designed to deal with high phase noise, when actual phase noise is low the algorithm exhibits good results, close to the conventional turbo code performance without phase noise.

The improved results shown in this paper demanded additional states and thus greater complexity. However, many sub-optimal methods with reduced complexity can be applied. In this article we have presented the method for BPSK modulation; however the same algorithm can be applied for other constellations as well.

This paper is organized as follows. After the system model is presented in Section II, we present the JCPETD algorithm in Section III and then the BCAPP decoder in Section IV. We derive the metric calculations of the BCAPP decoder in Section V, the component encoders selection in Section VI, and finally, simulation results are presented in Section VII followed by conclusions in Section VIII.

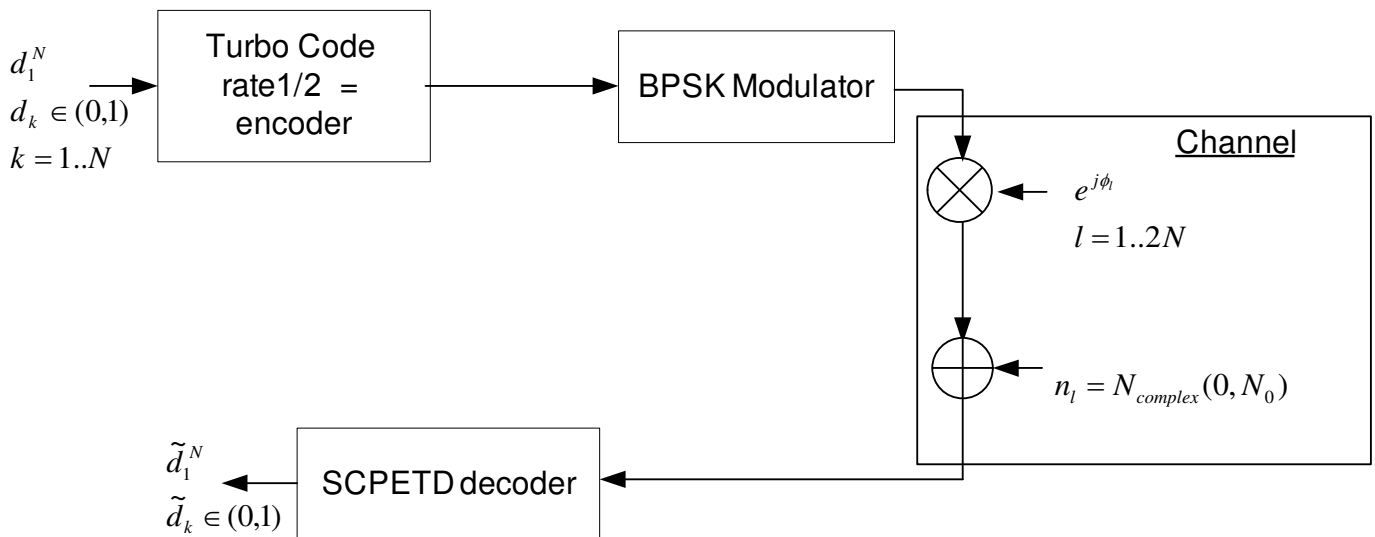


Fig. 1. JCPETD system block diagram.

II. SYSTEM MODEL

The communication system using the JCPETD algorithm, depicted in Fig. 1, consists of the turbo encoder, the channel, described in the following, and the JCPETD decoder, described in the next section.

A. The Turbo Encoder and Channel Model

In this article we use the typical Recursive Systematic Convolutional (RSC) turbo encoder of $Rate = 1/2$, presented by Berrou and Glavieux [15], illustrated in Fig. 2. The component encoders consist of two identical recursive convolutional encoders in parallel concatenation configuration and are separated by a pseudo-random interleaver. The component encoders have memory order of $\nu = 4$, feedback function represented by a polynomial G_1 , and an output function represented by a polynomial G_2 . The polynomials used for the encoder are optimized for the JCPETD needs as described in Section VI. The encoder encodes blocks of N information bits producing N systematic symbols, $d_1^N = (d_1, \dots, d_k, \dots, d_N)$, $d_k \in (0,1)$, $k = (1..N)$. Each of the two encoders, $ENC1$ and $ENC2$, produces N coded symbols that are punctured so as to produce $N/2$ coded symbols from each component encoder, $(c_1)_1^N$ and $(c_2)_1^N$ respectively. Unlike the coherent channel due to the phase dependencies, the order of the symbols cannot be arbitrary. We order the transmitted sequence as $(d_1, (c_1)_1, d_2, (c_2)_2, d_3, (c_1)_3, d_4, (c_2)_4, \dots, d_N, (c_2)_N)$.

The output of the encoder is BPSK modulated and transmitted over an AWGN channel, as illustrated in Fig. 1. Since we assume BPSK modulation of the bits $d_k, (c_1)_k, (c_2)_k \in (0,1)$, the transmitted symbols are represented by $(2d_k - 1)$, $(2(c_1)_k - 1)$ and $(2(c_2)_k - 1)$, respectively. The phase noise of the received samples is modeled by the Gaussian Random Walk (GRW) equation

$$\phi_l = \phi_{l-1} + (n_\phi)_l \quad l = 1..2N, \quad (1)$$

where n_ϕ is white Gaussian statistical independent random variable of variance σ_ϕ^2 and zero mean. Note that the phase noise is per symbol (and not per bit), thus the vector is of size $N/Rate = 2N$. Even with the high phase noise levels that we have in the channel, we can approximately assume that the phase change between the systematic symbol and the code symbol adjacent to it is zero. While such

assumption was made during the development of the algorithm, it was not used in the channel simulation. We denote the phase representing the two adjacent symbols of location k by θ_k , where $\theta_k = \phi_{2k}$, $k = 1..N$. Since ϕ_l is a first order Markov process and $\Pr(\phi_{l+1}|\phi_l) = N(0, \sigma_\phi^2)$, the conditional probabilities $\Pr(\phi_{2k+2}|\phi_{2k+1})$ and $\Pr(\phi_{2k+1}|\phi_{2k})$ are statistically independent. It is easy to show that the phase θ_k also behaves as a GRW process, but with variance $\sigma_\theta^2 = 2\sigma_\phi^2$.

The channel samples are denoted by $(x_k, (y_1)_k, (y_2)_k)$, where x_k , $(y_1)_k$ and $(y_2)_k$ are the complex samples of the received systematic symbol d_k , and the coded component punctured symbols $(c_1)_k$ and $(c_2)_k$, respectively. Finally, the sampled sequence is described as

$$(x_1, (y_1)_1, x_2, (y_2)_2, \dots, x_N, (y_2)_N) = ((2d_1 - 1) \cdot e^{j\phi_1} + n_1, (2(c_1)_1 - 1) \cdot e^{j\phi_2} + n_2, (2d_2 - 1) \cdot e^{j\phi_3} + n_3, (2(c_2)_2 - 1) \cdot e^{j\phi_4} + n_4, \dots, (2d_N - 1) \cdot e^{j\phi_{2N-1}} + n_{2N-1}, (2(c_2)_N - 1) \cdot e^{j\phi_{2N}} + n_{2N}).$$

Note that $x_k, (y_1)_k$ and $(y_2)_k$ were rotated by the channel, and the rotating phase is both unknown and fast time varying over the block.

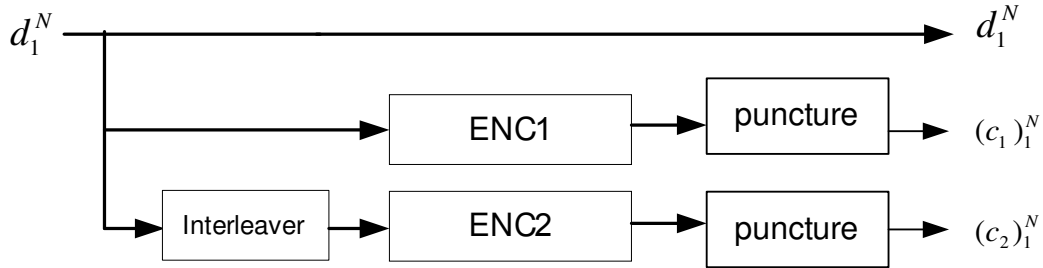


Fig. 2. Turbo Code encoder for $Rate = 1/2$.

III. THE TURBO RECEIVER DESIGN

Consider the conventional receiver for parallel concatenated turbo codes using the BCJR MAP decoder [6], where each BCJR MAP decoder is fed with coherent

real samples of the transmitted block and produces one type of extrinsic metric describing the bit value. In the JCPETD algorithm, shown in Fig. 3, each BCJR MAP decoder is replaced with a new bit-carrier-phase a posteriori probability (BCAPP) decoder that receives phase rotated complex samples of symbols and produces two separate extrinsic metrics, one for the bit and one for the carrier phase. The metric for the carrier phase is a vector rather than a single value, as will be explained in the following.

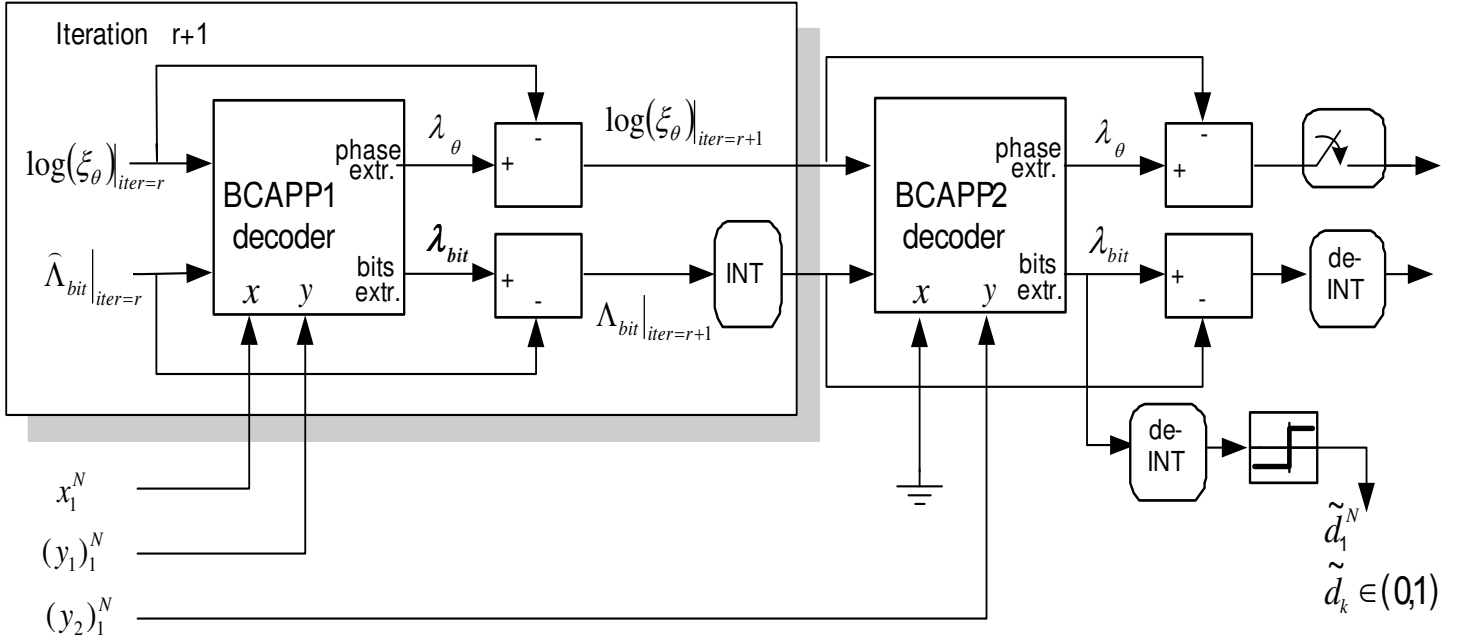


Fig. 3. JCPETD System block diagram for Rate = 1/2.

The inputs of the BCAPP decoder block are two complex samples per bit. The first is the systematic symbol, x_k , and the second is either $(y_1)_k$ or $(y_2)_k$ as depicted in Fig. 3. In the following, the input samples corresponding to the component code will be denoted generally by y_k and their corresponding encoder, $(c_1)_k$ or $(c_2)_k$, by c_k . The pairs of samples per bit are denoted by $R_k = (x_k, y_k)$ and the N pairs of samples received by each BCAPP are denoted by $R_1^N = (R_1, \dots, R_k, \dots, R_N)$.

The BCAPP decoder produces $\lambda_{bit,k}$ and $\lambda_{\theta,k}$, where $\lambda_{bit,k}$ describes the log likelihood ratio (LLR) for bit $d_k \in (0,1)$

$$\lambda_{bit,k} = \log \left(\frac{\Pr(d_k = 0 | R_1^N)}{\Pr(d_k = 1 | R_1^N)} \right), \quad (2)$$

where $\Pr(d_k = i | R_1^N)$ describes the A Posteriori Probability (APP) for bit d_k . We denote the bit value LLR vector by $\lambda_{bit} = (\lambda_{bit,1}, \dots, \lambda_{bit,k}, \dots, \lambda_{bit,N})$. λ_{bit} is also used for the hard decision of the bit value, denotes by \tilde{d}_k .

The continuous variable θ is practically quantized into P discrete values. Similar to the a posteriori data probabilities of the bit d_k , we calculate $\lambda_{\theta,k}(n)$, the a posteriori probability for its phase, to be equal to n , as follows (in the rest of the paper phases are measured in integer units of quantization steps or radians according to the context),

$$\lambda_{\theta,k}(n) = \log(\Pr(\theta_k = n | R_1^N)). \quad (3)$$

The values of $\lambda_{\theta,k}(n)$ are organized in a $N \times P$ matrix $\lambda_{\theta} = (\bar{\lambda}_{\theta,1}, \dots, \bar{\lambda}_{\theta,k}, \dots, \bar{\lambda}_{\theta,N})$, where $\bar{\lambda}_{\theta,k} = (\lambda_{\theta,k}(1), \dots, \lambda_{\theta,k}(n), \dots, \lambda_{\theta,k}(P))^T$.

Finally, as shown in Section V, the two *extrinsic* metrics are calculated by extracting the intrinsic information from $\lambda_{bit,k}$ and $\lambda_{\theta,k}$. The bit extrinsic metric, denoted by Λ_{bit} where $\Lambda_{bit} = (\Lambda_{bit,1}, \dots, \Lambda_{bit,k}, \dots, \Lambda_{bit,N})$, is produced by subtracting the intrinsic metric from the LLR metric,

$$\Lambda_{bit,k} \Big|_{iter=r+1} = \lambda_{bit,k} - \hat{\Lambda}_{bit,k} \Big|_{iter=r}, \quad (4)$$

where $\Lambda_{bit} \Big|_{iter=r+1}$ denotes the extrinsic metric for bit d_k after iteration $r+1$ and $\hat{\Lambda}_{bit} \Big|_{iter=r}$ denotes the intrinsic metric fed to iteration $r+1$ (The hat sign indicates an interleaved value). The phase extrinsic metric at the output of iteration r is $N \times P$ matrix $\xi_{\theta} \Big|_{iter=r} = (\bar{\xi}_{\theta,1}, \dots, \bar{\xi}_{\theta,k}, \dots, \bar{\xi}_{\theta,N}) \Big|_{iter=r}$, $\bar{\xi}_{\theta,k} = (\xi_{\theta,k}(1), \dots, \xi_{\theta,k}(n), \dots, \xi_{\theta,k}(P))^T$, where $\xi_{\theta,k}(n) \Big|_{iter=r}$, $n = 1..P$ describes the phase extrinsic metric for phase state n and bit d_k . In practice, we prefer to pass logarithmic values rather than the values themselves. As shown in Section V and depicted in Fig. 3, we calculate the extrinsic phase metrics for the next iteration by subtracting the intrinsic metric from the log probability of the phase by

$$\log(\xi_{\theta,k}(n)) \Big|_{iter=r+1} = \lambda_{\theta,k}(n) - \log(\xi_{\theta,k}(n)) \Big|_{iter=r}. \quad (5)$$

The new JCPETD decoder keeps two separate extrinsic metrics, one for the carrier-phase and one for the bits value. While the bits value extrinsic information is

interleaved and de-interleaved between the BCAPP decoders the carrier-phase extrinsic information remains always in a time-related sequential order. Since GRW phase noise is assumed, the conditional phase of a symbol is Gaussian distributed with mean equal to the phase of the previous symbol pair. As mentioned earlier, we assume the same phase for the systematic symbol and the code symbol adjacent to it. As a result, one phase extrinsic metric, denoted by $\xi_{\theta,k}(n)$, $n = 1..P$, represents the estimation of the received phase of x_k , $(y_1)_k$ and $(y_2)_k$. It is maybe possible to improve the performance of the algorithm by including the phase variations in the metrics calculations.

While the first BCAPP decoder, *BCAPP1*, receives the systematic samples, the component code samples and the intrinsic metrics, the second decoder, *BCAPP2*, does not receive the systematic samples as a complex input to the metric calculation. The reason is that the systematic samples lose their meaning if interleaved since their phase is unknown. The information conveyed in the systematic samples is not lost, however. It is separated by *BCAPP1* to the bit and phase extrinsic metrics and then transferred to the *BCAPP2* decoder. It can be interpreted as if *BCAPP1* de-rotates the systematic symbol by its current estimate of the phase. Therefore, separation of the systematic information to interleaved bit extrinsic information and uninterleaved phase extrinsic information, is essential.

IV. BCAPP DECODER BLOCK

The unmodified BCJR MAP [6] decoder, used for turbo decoding, contains 2^ν states, where ν is the number of taps in the RSC encoder's shift register. Let the time and encoder's state define a two-dimension matrix. With the BCAPP decoder it is convenient to define a third dimension. This dimension represents all the received (quantized) carrier phase, which will be referred to as the phase state. Each phase state represents one out of P equal angular parts dividing the 2π circle. In this paper the value of $P = 18$ is chosen so that quantization error of the phase will not result in more than 0.1dB SNR degradation. This degradation was computed approximately by assuming a coherent BSPK demodulator in the presence of a phase offset, distributed uniformly over the possible quantization error. This degradation is calculated by

$$Deg_{\theta} = \log(E[\cos^2(\theta)]) = 10 \cdot \log \left(\frac{P}{2\pi} \int_{-\frac{\pi}{P}}^{\frac{\pi}{P}} \cos^2(\vartheta) d\vartheta \right), \quad (6)$$

where $\Pr(\theta) = P/2\pi, \theta \in (-\pi/P, \pi/P)$.

Note that the description of the a three-dimension trellis, is just for convenience. The combination of the phase states and the code states can be viewed as super-states on augmented trellis.

The encoder used here is described in Section II-A and is the typically used RSC encoder with memory order of $\nu = 4$. The information bit d_k causes the change of the encoder state from state $S_k = m, m \in (0..(2^\nu - 1))$ at time k to state $S_{k+1} = f_s(m, d_k)$ at time $k+1$, where the function f_s is defined by the encoder feedback polynomial G_1 . In a similar way and at the same time, the encoder produces the symbols $c_k = c^{d_k, m}$ using the feed forward polynomial G_2 .

While f_s produces only two optional states at time $k+1$ for each state at time k , in the phase states axes every phase state can result from any other phase state at time k with a probability that is proportional to the phase difference. We will henceforth call these probabilities Phase Difference Probabilities (PDP). We will denote the phase difference probability as a transition from phase state $\theta_k = n$ at time k to phase $f_{\theta}(j, n)$ at time $k+1$ due to phase change of $2\pi j/P$ rad, $j \in (-(P/2 - 1)..(P/2))$, where $f_{\theta}(j, n) = \text{mod}(j + n, P)$, as $\Pr(d\theta_k = j)$.

The possible branches between states are depicted in Fig. 4. The probability of each branch, denoted by branch metric (BM), is equal to the probability of sample R_k to be a sample of transmitted bit $d_k = i$ when the encoder state is $S_k = m$, the receive phase is $\theta_k = n$ at time k and phase of $f_{\theta}(j, n)$ at time $k+1$ (due to phase change of $2\pi j/P$ rad).

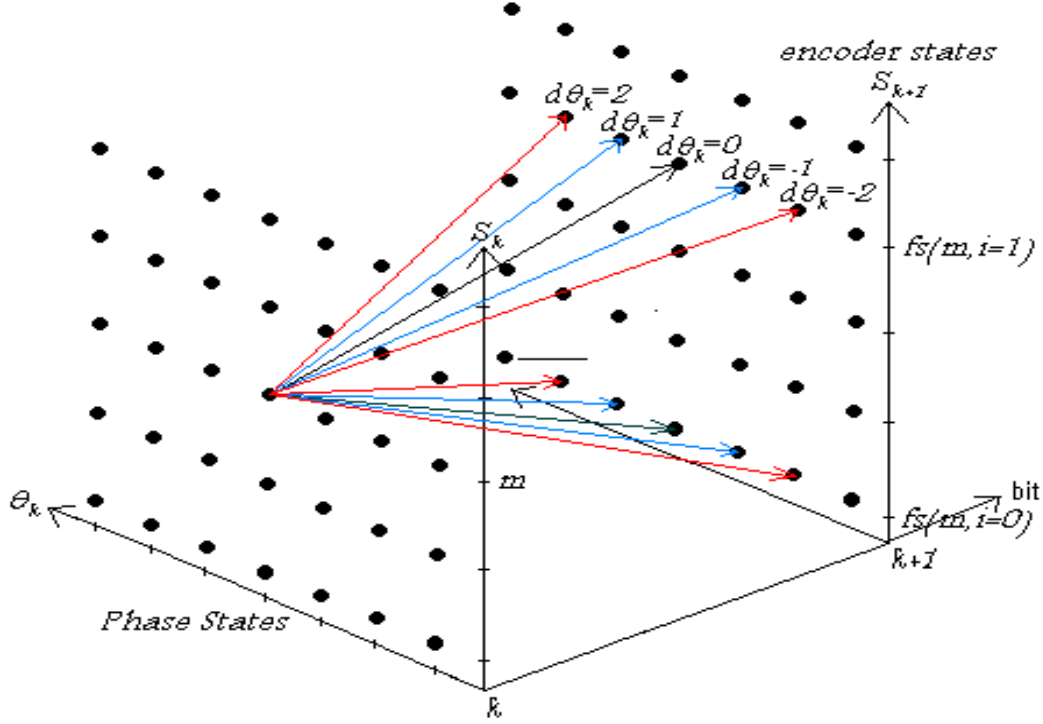


Fig. 4. Bit-Phase state diagram of the BCAPP decoder and the branch metrics

Since a zero mean GRW phase noise is assumed, the Phase Difference Probability (PDP) decreases as the magnitude of the phase change between adjacent symbols increases. We derive the PDP from the Gaussian distribution.

Under the assumption of GRW phase distribution, the conditional pdf of the phase representing bit k is

$$\Pr(\theta_k | \theta_{k-1}) = \frac{1}{\sqrt{2\pi\sigma_\theta^2}} \exp\left(-\frac{1}{2\sigma_\theta^2}(\theta_k - \theta_{k-1})^2\right). \quad (7)$$

For PDP calculation we assume that the phase of the previous symbol, $k-1$, represented by a certain phase state is in the middle of the phase range represented by that state. We then use Equ. (7) to calculate the PDP for a phase change of size j

$$\begin{aligned} \Pr(d\theta_k = j) &= \Pr[(\theta_k - \theta_{k-1}) \in (2\pi/P \cdot (j+1/2), 2\pi/P \cdot (j-1/2))] = \\ &= \int_{\frac{2\pi}{P}(j-1/2)}^{\frac{2\pi}{P}(j+1/2)} \frac{1}{\sqrt{2\pi\sigma_\theta^2}} \exp\left(-\frac{1}{2\sigma_\theta^2}g^2\right) dg \end{aligned} \quad (8)$$

All phases are folded into the $(-\pi:\pi)$ rad range. Note that due to the high resolution of phase quantization the exact method of calculating the PDP has a little effect.

For GRW phase noise with variance $\sigma_\phi \leq 0.323$ Rad, investigated in this paper, seven PDP values greater than zero showed no observable degradation (degradation ≤ 0.02 dB for BER = 10^{-3} after 10 iterations) compared to the maximum possible 18 PDP values greater than zero, therefore reducing needed calculation.

Calculating the PDP requires previous knowledge of the phase noise distribution, which in most cases is unknown. Using PDP assuming lower variance than the actual phase noise will result in poor phase estimation similar to the effect of narrowing the loop filter of a Phase Locked Loop (PLL). However, increasing the variance of the PDP adds more branches between states increasing sensitivity to noise (refer to Section VI). Methods for estimating the phase noise spectrum are not discussed in this paper, but results of mismatched phase noise variance are presented in the results section (Section VII) showing that the loss from pessimistic assumption regarding the phase noise level is small.

In Section V we derive the calculation of both bit and carrier phase metric. For reduced complexity, the suggested modified MAP block can easily be replaced with any other modified SISO (Soft In Soft Out) algorithm like SOVA (Soft Output Viterbi Algorithm).

V. DESIGN OF THE BCAPP

Based on the idea of BCJR MAP decoder, we derive the BCAPP decoder enhanced with the capability of the carrier phase estimation. The iterative turbo structure feeds the BCAPP decoder with the bit and phase a priori probabilities. We denote $\xi_{bit,k}^0, \xi_{bit,k}^1$ as the a priori probability for bit d_k to be '0' or '1' respectively ($\xi_{bit,k}^0 + \xi_{bit,k}^1 = 1$), and $\xi_{\theta,k}(n), n \in (1..P)$ to be the a priori probability for bit d_k to be at phase n . The a priori probabilities of the bits can be calculated from the intrinsic metric $\hat{\Lambda}_{bit,k}$ using

$$\widehat{\Lambda}_{bit,k} = \log \left(\frac{\xi_{bit,k}^0}{\xi_{bit,k}^1} \right) \quad (9)$$

We define log likelihood ratio (LLR) for information bit d_k

$$\lambda_{bit,k} = \log \left(\frac{\Pr(d_k = 0 | R_1^N)}{\Pr(d_k = 1 | R_1^N)} \right), \quad (10)$$

where $\Pr(d_k = i | R_1^N)$ describes the A Posteriori Probability (APP) for bit d_k .

The following joint probability is used to derive the a posteriori probabilities of both bit and phase

$$\psi_k^{i,m,j,n} = P(d_k = i, S_k = m, d\theta_k = j, \theta_k = n, R_1^N) \quad (11)$$

The joint probability $\psi_k^{i,m,j,n}$ represent the probability of bit $d_k = i, i \in (0,1)$, the encoder state $S_k = m, m \in (0 \dots (2^v - 1))$, the phase $\theta_k = n, n \in (1 \dots P)$ and the phase change from bit k to bit $k+1$ to be $d\theta_k = j, j \in (-(P/2 - 1) \dots (P/2))$ all together. Since we divide the full 2π circle into P sections, both n and j take discrete values.

The LLR for the data of bit d_k , Equation (10), can now be written as

$$\lambda_{bit,k} = \log \left(\frac{\sum_{m,j,n} \psi_k^{0,m,j,n}}{\sum_{m,j,n} \psi_k^{1,m,j,n}} \right). \quad (12)$$

The final (hard decision) results of the turbo decoder are obtained by comparing $\lambda_{bit,k}$ to 1.

Similar to the a posteriori data probabilities of the bit d_k we calculate the a posteriori phase probability for its phase to be n

$$\lambda_{\theta,k}(n) = \log \left(\Pr(\theta_k = n | R_1^N) \right) \quad (13)$$

or by using the joint probability

$$\lambda_{\theta,k}(n) = \log \left(\sum_{i,m,j} \psi_k^{i,m,j,n} \right). \quad (14)$$

In order to make the calculation implementable we will use Bayes' rule to rewrite the joint probability

$$\begin{aligned}
\psi_k^{i,m,j,n} &= \Pr(d_k, S_k = m, d\theta_k = j, \theta_k = n, R_1^N) \\
&= \Pr(R_1^{K-1} \mid d_k = i, S_k = m, d\theta_k = j, \theta_k = n, R_K^N) \cdot \\
&\quad \cdot \Pr(R_{K+1}^N \mid d_k = i, S_k = m, d\theta_k = j, \theta_k = n, R_K) \cdot \\
&\quad \cdot \Pr(d_k = i, S_k = m, d\theta_k = j, \theta_k = n, R_K)
\end{aligned} \tag{15}$$

Next we show how to calculate each of the three probabilities presented in Equ. (15). For the samples prior to d_k we can simplify the probability expression if we assume the phase to be a first order Markov process

$$\begin{aligned}
\Pr(R_1^{K-1} \mid d_k = i, S_k = m, d\theta_k = j, \theta_k = n, R_K^N) \\
&= \Pr(R_1^{K-1} \mid S_k = m, \theta_k = n) \\
&= \alpha_k^{m,n}.
\end{aligned} \tag{16}$$

Since the state m and the phase n are the only needed values to define the probability, we derive this simple expression. We define $\alpha_k^{m,n}$ as the **Forward State Metric**.

For the samples following d_k we use the same assumption about the phase

$$\begin{aligned}
\Pr(R_{k+1}^N \mid d_k = i, S_k = m, d\theta_k = j, \theta_k = n, R_k) \\
&= \Pr(R_{k+1}^N \mid S_{k+1} = f_S(i, m), \theta_{k+1} = f_\theta(j, n)) \\
&= \beta_{k+1}^{f_S(i,m), f_\theta(j,n)}
\end{aligned} \tag{17}$$

Here the samples are dependent on the encoder and phase states ($f_S(i, m), f_\theta(j, n)$ respectively) of the subsequent bit ($k+1$). The metric $\beta_{k+1}^{f_S(i,m), f_\theta(j,n)}$ is defined as the **Reverse State Metric**.

The third probability in Equ. (15) is defined as the branch metric

$$\begin{aligned}
& \Pr(d_k = i, S_k = m, d\theta_k = j, \theta_k = n, R_k) \\
&= \Pr(d_k = i, S_k = m, \theta_k = n, R_k) \cdot \Pr(d\theta_k = j) \\
&= \delta_k^{i,m,j,n}
\end{aligned} \tag{18}$$

The branch metric calculates the probability of sample R_k to be of bit $d_k = i$, state $S_k = m$ and received in phase $\theta_k = n$. The branch metric includes the phase difference probability to pass from phase $\theta_k = n$ at time k to phase $f_\theta(j, n)$ at time $k+1$ due to phase change of $2\pi j/P$ Rad.

Finally, placing Equ. (16), (17) and (18) into (15), the joint probability used for deriving the LLR of the bit and log probability of the phase can be rewritten as

$$\psi_k^{i,m,j,n} = \alpha_k^{m,n} \delta_k^{i,m,j,n} \beta_k^{f_S(i,m), f_\theta(j,n)} \tag{19}$$

Similar to the functions $f_S(i, m), f_\theta(j, n)$, that calculate the next encoder state and phase state, respectively, we denote by $b_S(i, m), b_\theta(j, n)$ the functions that calculate the preceded encoder state and phase state, respectively. These functions are needed for the Forward State metric calculations. Next we show how to calculate the Forward and Backward Metrics.

In order to make the calculation recursive, the Forward State metrics (16) can be written as

$$\begin{aligned}
\alpha_k^{m,n} &= \Pr(R_1^{k-1} | S_k = m, \theta_k = n) \\
&= \sum_{i,m',j,n'} \Pr(R_1^{k-1}, d_{k-1} = i, S_{k-1} = m', d\theta_{k-1} = j, \theta_{k-1} = n' | S_k = m, \theta_k = n) \\
&= \sum_{i=0}^1 \sum_{m',j,n'} \left[\Pr(R_1^{k-2} | d_{k-1} = i, S_{k-1} = m', d\theta_{k-1} = j, \theta_{k-1} = n', S_k = m, \theta_k = n, R_{k-1}) \cdot \right. \\
&\quad \left. \Pr(R_{k-1}, d_{k-1} = i, S_{k-1} = m', d\theta_{k-1} = j, \theta_{k-1} = n' | S_k = m, \theta_k = n) \right] \\
&= \sum_{i=0}^1 \sum_{j=1}^p \left[\Pr(R_1^{k-2} | S_{k-1} = b_S(i, m), \theta_{k-1} = b_\theta(j, n)) \right. \\
&\quad \left. \Pr(R_{k-1}, d_{k-1} = i, S_{k-1} = b_S(i, m), d\theta_{k-1} = j, \theta_{k-1} = b_\theta(j, n)) \right] \\
&= \sum_{i=0}^1 \sum_{j=1}^p \alpha_{k-1}^{b_S(i,m), b_\theta(j,n)} \cdot \delta_{k-1}^{i, b_S(i,m), j, b_\theta(j,n)}
\end{aligned} \tag{20}$$

The same can be done for the Reverse Metric presented in Equ. (17)

$$\begin{aligned}
\beta_k^{m,n} &= \Pr(R_k^N | S_k = m, \theta_k = n) \\
&= \sum_{i,m',j,n'} \Pr(R_k^N, d_k = i, S_{k+1} = m', d\theta_k = j, \theta_{k+1} = n' | S_k = m, \theta_k = n) \\
&= \sum_{i=0}^1 \sum_{m',j,n'} \left[\Pr(R_{k+1}^N | d_k = i, S_{k+1} = m', d\theta_k = j, \theta_{k+1} = n', S_k = m, \theta_k = n, R_k) \cdot \right. \\
&\quad \left. \Pr(R_k, d_k = i, S_{k+1} = m', d\theta_k = j, \theta_{k+1} = n' | S_k = m, \theta_k = n) \right] \\
&= \sum_{i=0}^1 \sum_{j=1}^p \left[\Pr(R_{k+1}^N | S_{k+1} = f_S(i, m), \theta_{k+1} = f_\theta(j, n)) \cdot \Pr(R_k, d_k = i, S_k = m, d\theta_k = j, \theta_k = n) \right] \\
&= \sum_{i=0}^1 \sum_{j=1}^p \beta_{k+1}^{f_S(i,m), f_\theta(j,n)} \cdot \delta_k^{i,m,j,n}
\end{aligned} \tag{21}$$

The branch metric $\delta_k^{i,m,j,n}$ calculation includes probabilities of the input samples $R_k = (x_k, y_k)$, the PDP $\Pr(d\theta_k = j)$ and the intrinsic probabilities of the state and the phase, $\xi_{bit,k}^i, \xi_{\theta,k}^n$, respectively. The branch metric described in Equation (18) can be written as follows.

$$\begin{aligned}
\delta_k^{i,m,j,n} &= \Pr(d_k = i, S_k = m, d\theta_k = j, \theta_k = n, R_k) \\
&= \Pr(d_k = i, S_k = m, \theta_k = n, R_k) \cdot \Pr(d\theta_k = j) \\
&= \Pr(R_k | d_k = i, S_k = m, \theta_k = n) \cdot \\
&\quad \Pr(S_k = m | d_k = i, \theta_k = n) \cdot \Pr(d_k = i, \theta_k = n) \cdot \Pr(d\theta_k = j) \\
&= \frac{\Pr((x_k, y_k) | d_k = i, S_k = m, \theta_k = n) \cdot \Pr(d_k = i)}{2^\nu} \\
&\quad \cdot \Pr(\theta_k = n) \cdot \Pr(d\theta_k = j) \\
&= \frac{\Pr(x_k | d_k = i, \theta_k = n) \cdot \Pr(y_k | d_k = i, S_k = m, \theta_k = n)}{2^\nu} \\
&\quad \cdot \xi_{bit,k}^i \xi_{\theta,k}^n \Pr(d\theta_k = j)
\end{aligned} \tag{22}$$

Since an AWGN channel is assumed, the following probabilities, which are proportional to the Euclidian distance from the supposed constellation symbol, can be obtained.

$$\begin{aligned}
\Pr(R_k \mid d_k = i, S_k = m, \theta_k = n) &= \\
&= \Pr(x_k \mid d_k = i, S_k = m, \theta_k = n) \cdot \Pr(y_k \mid d_k = i, S_k = m, \theta_k = n) \\
&= \frac{1}{2\pi\sigma^2} \exp\left(-\frac{1}{2\sigma^2} \left| x_k - (2i-1) \cdot \exp(j \frac{2\pi}{P} n) \right|^2\right) \\
&\quad \cdot \exp\left(-\frac{1}{2\sigma^2} \left| y_k - (2c^{i,m} - 1) \cdot \exp(j \frac{2\pi}{P} n) \right|^2\right) \\
&= \text{Const} \cdot \exp\left(\frac{1}{\sigma^2} \text{Re}\left(x_k \cdot (2i-1) \cdot \exp(-j \frac{2\pi}{P} n)\right)\right) \\
&\quad \cdot \exp\left(\frac{1}{\sigma^2} \text{Re}\left(y_k \cdot (2c^{i,m} - 1) \cdot \exp(-j \frac{2\pi}{P} n)\right)\right)
\end{aligned} \tag{23}$$

The branch metric can be summarized as

$$\delta_k^{i,m,j,n} = \xi_{bit,k}^i \xi_{\theta,k}^n \Pr(d\theta_k = j) \Pr(R_k \mid d_k = i, S_k = m, \theta_k = n) \tag{24}$$

Substituting the branch metric, Equ. (24), into Equ. (12) and Equ. (14) produces

$$\begin{aligned}
\lambda_{bit,k} &= \log \left(\frac{\sum_{m,j,n} \psi_k^{0,m,j,n}}{\sum_{m,j,n} \psi_k^{1,m,j,n}} \right) \\
&= \log \left(\frac{\xi_{bit,k}^0 \sum_{m,j,n} \alpha_k^{m,n} \beta_{k+1}^{f_S(0,m), f_\theta(j,n)}}{\xi_{bit,k}^1 \sum_{m,j,n} \alpha_k^{m,n} \beta_{k+1}^{f_S(1,m), f_\theta(j,n)}} \right) \\
&\quad \cdot \frac{\Pr(R_k \mid d_k = 0, S_k = m, \theta_k = n) \Pr(d\theta_k = j) \cdot \xi_{\theta,k}^n}{\Pr(R_k \mid d_k = 1, S_k = m, \theta_k = n) \Pr(d\theta_k = j) \cdot \xi_{\theta,k}^n}
\end{aligned} \tag{25}$$

$$\begin{aligned}
\lambda_{\theta,k}(n) &= \log \left(\sum_{i,m,j} \psi_k^{i,m,j,n} \right) \\
&= \log \left(\xi_{\theta,k}^i(n) \sum_{i,m,j} \alpha_k^{i,m,n} \beta_{k+1}^{f_S(i,m), f_\theta(j,n)} \Pr(R_k \mid d_k = i, S_k = m, \theta_k = n) \Pr(d\theta_k = j) \xi_{bit,k}^i \right)
\end{aligned} \tag{26}$$

Notice that the state and phase intrinsic probabilities can be factored out, avoiding feeding them back to the subsequent decoder that produced them during the previous

iteration. The extrinsic metrics are produced by subtracting the intrinsic bit and phase metrics from the bit log likelihood probabilities and phase log probabilities, respectively, as follows

$$\Lambda_{bit,k} \Big|_{iter=r+1} = \lambda_{bit,k} - \widehat{\Lambda}_{bit,k} \Big|_{iter=r} \quad (27)$$

$$\log\left(\xi_{\theta,k}(n)\right) \Big|_{iter=r+1} = \lambda_{\theta,k}(n) - \log\left(\xi_{\theta,k}(n)\right) \Big|_{iter=r} \quad (28)$$

However, it is not possible to factor out all of the intrinsic probabilities from the extrinsic probabilities since there exist a cross connection between the two probabilities as one can see from Equ. (25) and (26). For example, the bit likelihood probability $\lambda_{bit,k}$ depends on $\xi_{\theta,k}^n$ in a manner that is hard to factor out. This cross connection is discussed in the performance and result section (VII).

VI. OPTIMIZING THE ENCODER

Let us describe the component code encoder as *ENCI* with input information bits $d_1^N = (d_1, d_2, \dots, d_N), d_k \in (0,1)$. We note the functionality of the encoder as $ENCI(d_1^N) = c_1^N$ where $c_1^N = (c_1, c_2, \dots, c_N) \in C, c_k \in (0,1)$ is any codeword in the *ENCI* codeword space described by C . Let us define the following description for a trellis $[d_1^N, c_1^N, \theta_1^N] \equiv [(d_1, d_2, \dots, d_N), (c_1, c_2, \dots, c_N), (\theta_1, \theta_2, \dots, \theta_N)]$ where $\theta_1^N = (\theta_1, \theta_2, \dots, \theta_N)$ denote the trellis decoded phase.

We will define BPSK Rotational Invariant (RI) encoder as RSC encoder, where inversion of the information bits, described by $\overline{d_1^N} = (\overline{d_1}, \overline{d_2}, \dots, \overline{d_N})$, results in $ENCI(\overline{d_1^N}) = \overline{c_1^N}$ where $\overline{c_1^N} = (\overline{c_1}, \overline{c_2}, \dots, \overline{c_N}) \in C$. This represents a π Rad invariant rotation for code *ENCI*. In the RI encoder two trellises can coexist: the first trellis is described by $[d_1^N, c_1^N, \theta_1^N]$, and the second trellis is described by $[\overline{d_1^N}, \overline{c_1^N}, \overline{\theta_1^N}]$, where $\overline{\theta_1^N} = (\overline{\theta_1}, \overline{\theta_2}, \dots, \overline{\theta_N}), \overline{\theta_k} = \theta_k + \pi$. Since for the calculation of the bit LLR we sum up all of the branches for bit 1 and divide by the branches for bit 0, the two RI trellises will give inverted branches that when summed together will result in producing extrinsic metric with no information. Therefore, for the JCPETD decoder, the RI encoder is defined as Non-coherent Catastrophic (NC), see [16] Section III, and should be avoided. However, choosing a rotational variant encoder is not enough, since the

performance depends on the hamming distance of coded-symbols inverted sequences in an intricate manner.

As described in [17] Section II, turbo code bit error probability is mainly affected by the distance spectrum of the code especially at low SNR. The distance spectrum of a code considers the minimum Hamming distance of the codewords and their error coefficients (weight). The upper bound of the turbo code bit error probability is described by

$$\Pr(\text{error}) \leq \sum_{d=d_{\min}}^n A_d Q\left(\sqrt{d \cdot 2\text{Rate} \cdot E_b/N_0}\right) \quad (29)$$

where *Rate* is the code rate, E_b/N_0 is the ratio of the bit energy to noise power spectral density, d_{\min} is the minimum Hamming distance and A_d is the error coefficient which determines the contribution of the codewords with the same weight d to the bit error probability.

When adding the phase estimation, more trellises are possible for the decoder hence changing the distance spectrum. For example, let us consider the following two trellises

$$\left[d_1^N, c_1^N, \theta_1^N \right] \equiv \left[(d_1, d_2, \dots, d_N), (c_1, c_2, \dots, c_N), (\theta_1, \theta_2, \dots, \theta_N) \right]$$

and

$$\left[\hat{d}_1^N, \hat{c}_1^N, \hat{\theta}_1^N \right] \equiv \left[(d_1, \dots, \bar{d}_k, \dots, \bar{d}_j, \dots, d_N), (c_1, \dots, \bar{c}_k, \dots, \bar{c}_j, \dots, c_N), (\theta_1, \dots, \bar{\theta}_k, \dots, \bar{\theta}_j, \dots, \theta_N) \right].$$

The second trellis is a semi-inverted trellis of the first one and is a new trellis that only now becomes possible since the decoder estimates the phase. The option of phase inversion (π Rad change) adds many more trellises that change the distance spectrum and requires a modification of the RSC encoder. From simulation results it is evident that the affect of trellises like $\left[\hat{d}_1^N, \hat{c}_1^N, \hat{\theta}_1^N \right]$ on the bit error rate and the variance of the phase estimation can be crucial if the encoder is not carefully chosen, especially for low SNR and high phase noise. In the simulation we observed that the cases when phase inversion occurred along the block, the bit error rate was drastically higher.

Since we do not have a good analytical tool to estimate the performance of the turbo code system, we use an RSC encoder with two polynomials that were selected

empirically by simulating all possibilities and picking the polynomials with lower bit error rate. Analytical tools for performance prediction are subject of our future research. Since $v = 4$, we have a maximum of 2^4 possibilities for the feedback, times 2^5 for the feed forward, resulting in 512 possibilities. Many of the 512 possibilities can be cancelled out since they are RI. Each possibility of RSC encoder was tested with random interleavers, those that showed better bit error rate at high phase noise and low SNR were then tested more extensively. The method of picking better encoders can be improved using methods similar to those presented in [17].

VII. PERFORMANCE AND RESULTS

We present simulation results for BPSK turbo decoding over an AWGN channel using GRW phase noise or zero phase noise. The simulations use turbo code block length of 4000 bits, $Rate = 1/2$, with GRW phase noise variance of $\sigma_\phi = 0.087$ rad (5 deg), 0.21 rad (12 deg) and 0.323 rad (18.5 deg). The turbo encoder polynomials were chosen to be $G_1 = 35$ and $G_2 = 33$ (octal) as described in Section VI and a random interleaver was picked for every block simulated.

In theory if no phase noise exists, the channel capacity can be achieved using SNR of about 0 dB for rate 1/2. The typical BCJR MAP turbo decoder achieves bit error rate close to zero for SNR of about 1 dB when using the same parameters tested in this paper. As can be seen from Fig. 5, the greater the phase noise variance - the greater the coding degradation. This can be explained by the understanding that greater phase noise adds more noise, which results in more errors, similar to decreased SNR. Another effect concerns the number of minimum points and their distance spectrum in the Euclidean space. When adding the carrier phase dimension to the problem of detection, more minimum points (possible trellises) are added in the Euclidean space, and the spectrum of distances between codewords is worsened. Examples of new trellises that add minimum points in the Euclidean space are the semi-inverted trellises described in diction V-A. From simulation results it is evident that these phase inversion trellises govern the distance spectrum. Higher phase noise demands matching PDP with greater

variance, which results in more minimum points to be added which decreases the Euclidean minimum distance of the code.

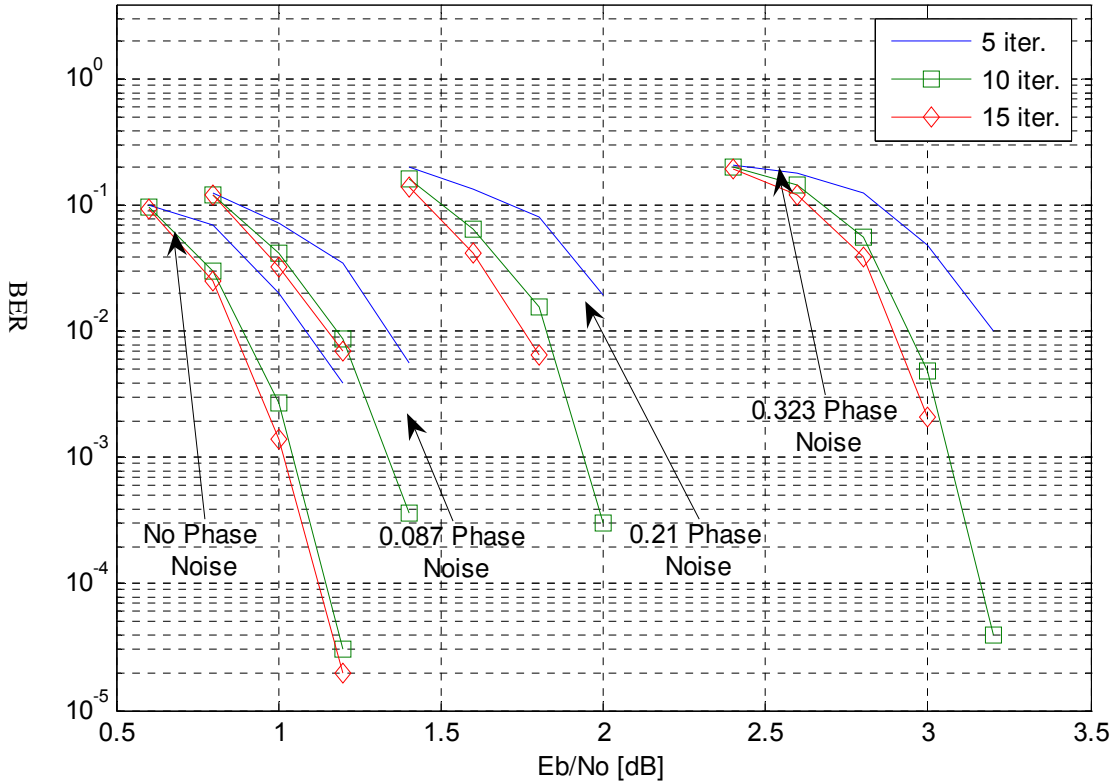


Fig. 5. Performance of the BPSK $Rate = 1/2$ Turbo Coded, $N = 4000$ bits per block over an AWGN channel with no phase noise and GRW phase noise of $\sigma_\phi = 0.087$ rad (5 deg), 0.21 rad (12 deg) and 0.323 rad (18.5 deg). Presented are the results of iterations 5, 10 and 18.

The results presented in Fig. 5 can be compared to the work of Peleg, Shamai and Galán in [1], where a MD (Modulation Detector) was concatenated with the turbo decoder to iteratively produce phase synchronization and turbo decoding. In the first iteration, a non-optimal operation of the MD results in higher SNR required for the BER to start to converge. The MD does not use the information conveyed in the coding for phase estimation therefore feeding the turbo decoder with a non-optimal intrinsic inputs. This is explained in [1, Section 3.2.2], as a non-optimal use of the Average Mutual Information (AMI) by the MD during the first iteration. The algorithm presented in this paper is better optimized due to the decoding of the phase and the bits simultaneously even at the first iteration, resulting in lower SNR needed to achieve low

BER ($<10^{-2}$). Observing Fig. 5, one can see that for $\sigma_\phi = 0.323$ Rad after 10 iterations, $E_b/N_0 = 3.05$ dB results in $BER = 10^{-3}$ which is 0.5 dB better than the results presented [1, Fig. 5].

Comparing the results of the method presented here to the MSDD (Multiple Symbol Differential Detection) system presented by Vainappel, Hardy and Raphaeli in [10], one can see that for $\sigma_\phi = 0.21$ Rad (12 Deg) the coherent method presented in this paper achieves about 0.6 dB gain for BER of 10^{-3} (compared with $L = 3$).

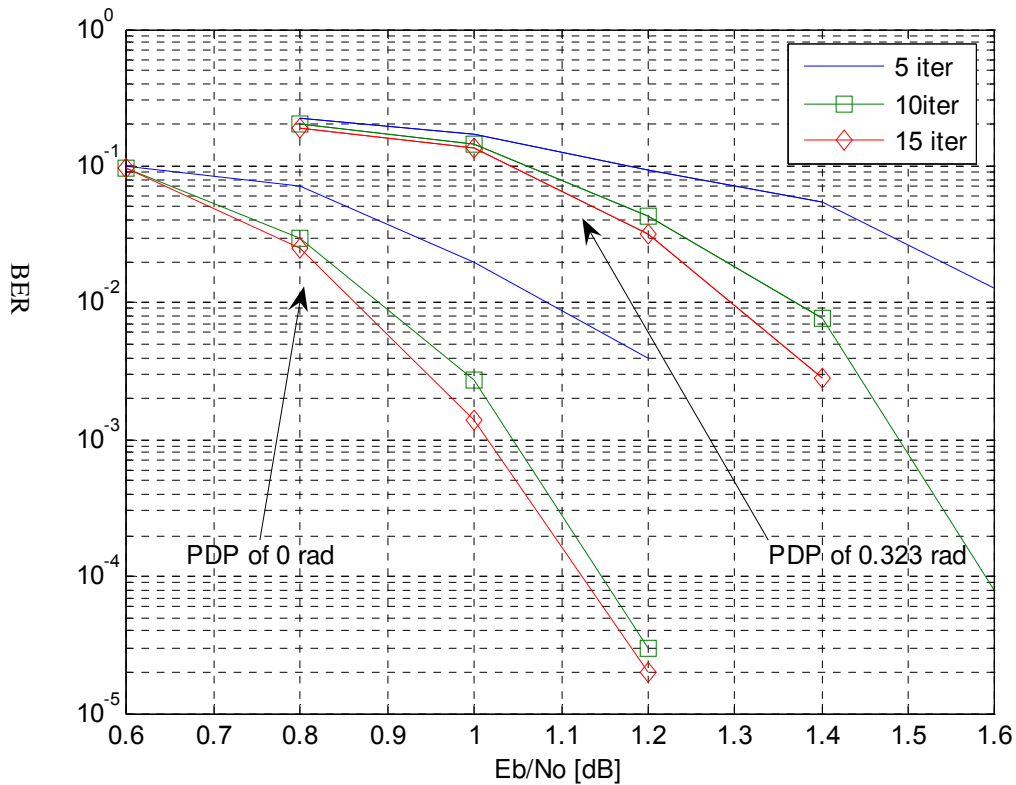


Fig. 6. Performance of the BPSK $Rate = 1/2$ Turbo Coded, $N = 4000$ bits per block over an AWGN channel with no actual phase noise and PDP set for phase noise variance of $\sigma_\phi = 0.323$ rad (18.5 deg) vs. $\sigma_\phi = 0$ rad.

Using the correct PDP is important to achieve the best results. However, knowing the phase noise characteristic of the received samples is not always possible and may vary in time and temperature. Fig. 6 shows that when there exist no phase noise but the

PDP parameters of the decoder are set for $\sigma_\phi = 0.323$ Rad, the degradation compared to regular decoding without phase estimation is about 0.45 dB for BER = 10^{-3} after 10 iterations. When compared with the results of zero phase noise presented by [1, Fig. 5], one can see that the results of the suggested algorithm in this paper are about 1.1dB better after 10 iterations for BER = 10^{-3} .

Due to the separation between the extrinsic information of the carrier phase and the extrinsic information of the data bits, we require the two to be also statistically independent. However, due to the structure of the BCAPP decoder, total separation is impractical; hence, there exist residual dependency between them as shown at the end of Section V, Equ. (25) and (26). When strong dependency was allowed, quick convergence of the phase and bit was achieved in earlier iterations. In some cases one parameter (usually the phase) reached high assurance (high probabilities of a specific state) quicker than the other parameter, resulting in most cases in converging to a wrong phase trellis. Best results were achieved by opening the feedback of the phase extrinsic information from the output of the second decoder into the first decoder, (depicted in Fig. 3 using a switch). As a result, we can avoid extracting the phase extrinsic information during the second half of each iteration.

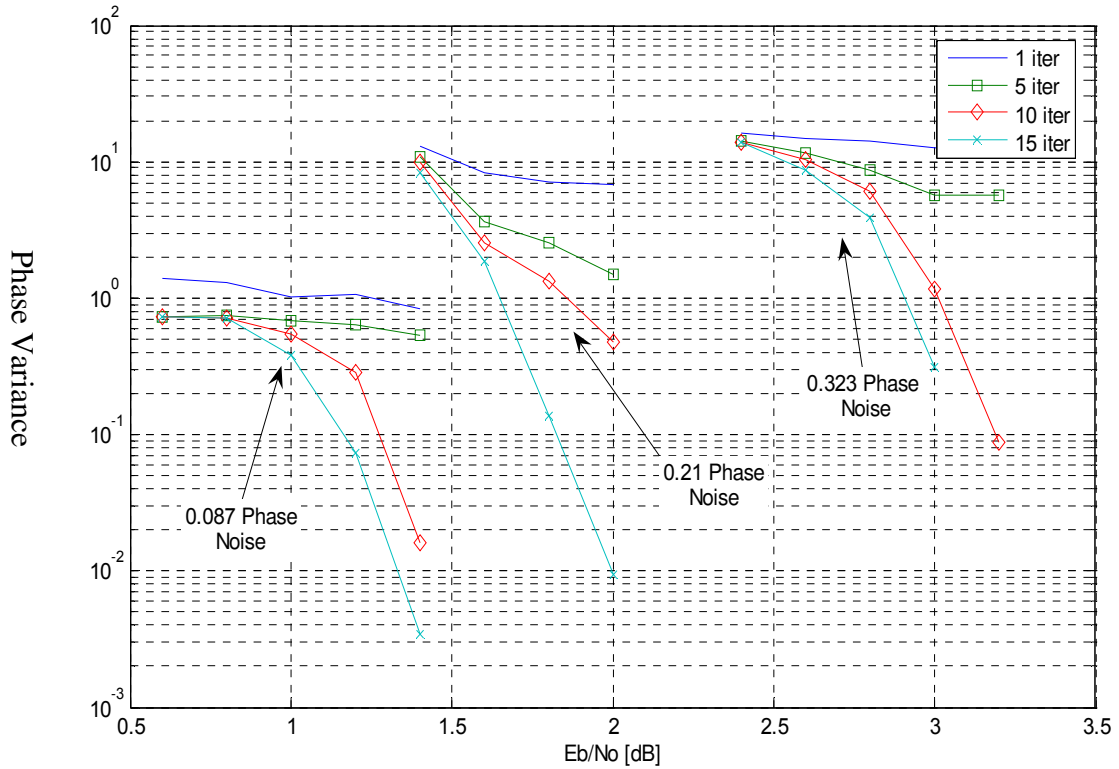


Fig. 7. Variance of the phase estimation of the BPSK $Rate = 1/2$ Turbo Coded, $N = 4000$ bits per block over an AWGN channel, with GRW phase noise of $\sigma_{\phi} = 0.087$ rad (5 deg), 0.21 rad (12 deg) and 0.323 rad (18.5 deg). Presented are the results of the output of the first BCAPP decoder in iterations 1, 5, 10 and 15.

The Phase noise variance presented in Fig. 7 represents the variance of the phase estimation probability. The phase estimation is converging jointly and simultaneously with the BER, as depicted in Fig. 5, indicating that both parameters jointly dictate the final results.

When recursively calculating the forward and backward metrics in a typical turbo decoder algorithm, it is preferable to know the exact state of the encoder at the start and at the end of the decoded data block. However, we can't assume such knowledge for the phase state. Fortunately, for block size of $N = 4000$ bits, no observable degradation was measured (degradation < 0.05 dB) when the first and last bit's phase states,

$\xi_{\theta,1}^n, \xi_{\theta,N}^n, n = 1..P$, were assumed to be unknown, i.e. uniform distributed over P states. Several SNR and phase noise combinations used in Fig. 4 were tested with and without knowledge of the first and last phase state, and showed similar BER performance, suggesting that it is not critical to know the first and last phase states.

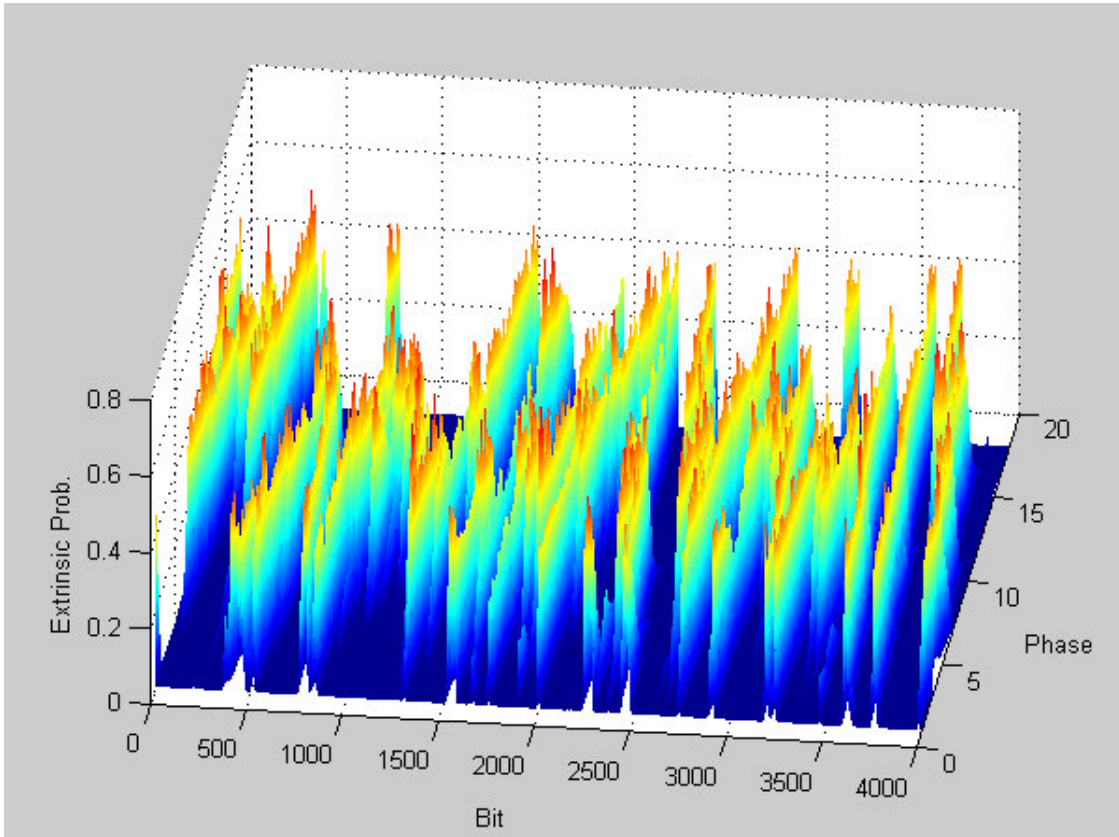


Fig. 8. Extrinsic Probability at the output of the 3rd iteration for phase noise of $\sigma_\phi = 0.17$ rad. $N = 4000$ bits per block and the phase axes of 1 to 18 phase states each represents $2\pi/18$ rad state.

In Fig. 8 we present an example of the extrinsic probabilities passed from the third iteration to the forth when decoding the GRW carrier phase with $\sigma_\phi = 0.17$ Rad.

VIII. CONCLUSION

The JCPETD iterative algorithm, which enables the estimation of the carrier phase and the data bit jointly, was introduced. The JCPETD algorithm is based on the BCAPP decoder, which is a modification of the BCJR MAP decoder with additional states representing the carrier phase. The BCAPP calculates the phase estimation and bit metric using two separate intrinsic streams of information for the phase and the data bit. It also produces two separate extrinsic streams used by the subsequent BCAPP decoder. The JCPETD algorithm achieves near optimum results and improved performance when carrier phase noise is either known or unknown. The algorithm demands additional states and thus greater complexity, but, on the other hand, achieves very good performance in high phase noise compared to previous algorithms, and also robustness against phase noise model mismatch. Sub-optimal methods requiring reduced complexity can also be applied. In this paper we have presented the method for BPSK detection; however the same algorithm can be applied for any other constellations. The concept of this paper can be extended to include the estimation of other variables of the channel as long as these variables can be represented as a Markov process. The option of expanding the estimation to other variables can be regarded as adding dimensions to the BCAPP decoder resulting in additional complexity and simple addition to the turbo iterative structure of the JCPETD algorithm.

REFERENCES

- [1] M. Peleg, S. Shamai (Shitz) and S. Galán, "Iterative Decoding for Coded Noncoherent MPSK Communications over Phase-Noisy AWGN Channel", *IEE Proc. Commun.*, Vol. 147, No. 2, pp. 87-95, Apr. 2000.
- [2] Wangrok Oh and Kyungwhoon Cheun, "Joint Decoding and Carrier Phase Recovery Algorithm for Turbo Codes", *IEEE Commun. Letters*, vol. 5, No. 9, pp. 375-377, Sept. 2001.
- [3] I. Motedayen and A. Anastasopoulos, "Polynomial-Complexity Noncoherent Symbol-by-Symbol Detection with Application to Adaptive Iterative Decoding of Turbo-Like Codes", *IEEE Trans. Commun.*, vol. 51, No. 2, pp. 197-207, Feb. 2003.

- [4] B. Mielczarek and A. Svensson, "Phase Offset Estimation using Enhanced Turbo Decoders", *IEEE International Conf. on Commun.*, ICC 2002, New York, USA, pp. 1536-1540. April 28-May 2, 2002.
- [5] I Bar-David and A. Elia, "Augmented APP (A^2P^2) module for a Posteriori Probability Calculation and Channel Parameter Tracking", *IEEE Commun. Letters*, vol. 3, No. 1, pp. 428-432, Jan. 1999.
- [6] L.Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", *IEEE Trans. Inform. Theory*, vol. IT-20, pp.284-287, Mar. 1974.
- [7] L. Zhang and A. G. Burr, "A New Method of Carrier Phase Recovery for BPSK System Using Turbo Codes over AWGN Channel", *PIMRC 2001*, Vol. 1, 30 Sept.-3 Oct. 2001 pp. 179-183.
- [8] C. Morlet, I. Buret, and M.L. Boucheret, "A carrier phase estimator for multimedia satellite payloads suited to RSC coding schemes," *ICC 2000*, Vol. 1, pp. 455-459, Jun. 2000.
- [9] Wilson, S.G and HSU, C.D., "Joint MAP data/phase sequence estimation for trellis phase codes", *ICC 1980*, pp. 26.1.1-6.1.5.
- [10] J. Vinappel, E. Hardy, and D. Rephaeli, "Noncoherent Turbo Decoding", *Globecom 2001*, San Antonio, Vol. 2, 25-29 Nov. 2001, pp. 952 - 956.
- [11] D. Rephaeli, "Noncoherent Coded Modulation", *IEEE Trans. On Commun.*, vol. 44, pp.172-183, Feb. 1996.
- [12] Hoehner, and Lodge, "Turbo DPSK: Iterative Differential PSK Demodulation and Channel Decoding", *IEEE Trans. On Commun.*, vol. 47, No. 6, pp. 837 – 843, June 1999.
- [13] Chayat, N, "Turbo Codes for Incoherent M-ary Orthogonal Signaling", *Proc., 19th Convntion of Electrical and Electronics Engineers in Israel*, pp. 471-4, 560, 1996.
- [14] M. Peleg, S. Shamai (Shitz), "On the Capacity of the Blockwise Incoherent MPSK Channel", *IEEE Trans. On Commun.*, vol. 46, No. 5, May 1998.
- [15] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes", *Proc. ICC 93*, Geneva, pp. 1064-1070, May 1993.
- [16] D. Rephaeli, G. Sittou and I. Taler "Geometrically Uniform Trellis Codes for Noncoherent Detection", *IEEE Trans. Commun.*, vol. 3, pp. 2182-2188, Nov. 2004.
- [17] J. Yuan, B. Vucetic and W. Feng "Combined Turbo Codes and Interleaver Design", *IEEE Trans. Commun.*, vol. 47, pp. 484-487, Apr. 1999.