Adaptive Iterative Detection for Phase Tracking in Turbo-Coded Systems

Achilleas Anastasopoulos, Member, IEEE, and Keith M. Chugg, Member, IEEE

Abstract—The problem of performing iterative detection (ID)—a technique originally introduced for the decoding of turbo codes—for systems having parametric uncertainty has received relatively little attention in the open literature. In this paper, the problem of adaptive ID (AID) for serial and parallel concatenated convolutional codes (SCCCs and PCCC's or turbo codes) in the presence of carrier-phase uncertainty is examined. Based on the theoretical framework in [1], [2], adaptive soft inverse (ASI) algorithms are developed for two commonly used blocks in turbo codes, leading to the adaptive soft-input soft-output (A-SISO) and the adaptive soft demodulator (A-SODEM) algorithms. Based on these algorithms, practical AID receivers are presented. Several design options are proposed and compared and the impact of parametric uncertainty on previously established results for iterative detection with perfect channel state information (CSI) is assessed.

I. INTRODUCTION

THE introduction of parallel concatenated convolutional codes (PCCC's or turbo codes) [3], and serial concatenated convolutional codes (SCCC's) [4] was arguably one of the most significant advances in coding theory. These codes are constructed as concatenations of simple constituent codes, and have been shown to achieve near-capacity performance. In a standard SCCC, shown in Fig. 1(a), a block of the input sequence is encoded by an outer convolutional code (CC). The coded symbols are fed to an inner CC through a pseudorandom interleaver and the output symbols are then mapped onto the constellation points and transmitted to the channel.

When perfect channel state information (CSI) is available, a decoder that approximates maximum-likelihood sequence detection (MLSD) performance with reasonable complexity can be constructed, by utilizing a novel decoding technique, namely iterative detection (ID) [3]. The effectiveness of ID can be attributed to the exchange of soft information related to the input/output symbols of each constituent CC. In [5], a systematic approach to designing ID receivers was presented for systems consisting of an interconnection of multiple subsystems. In this approach, which is consistent with the more general theory of probability propagation in graphs [6], soft inverse blocks—each one performing optimal processing considering only the local structure of the corresponding subsystem in the original network—are connected in an intuitive way to form the ID network. In general, the soft inverse requires marginalization of joint probabilities over all combinations of possible input/output sequences. When the subsystem is a finite state machine (FSM), however, the soft inverse can be computed efficiently using the so-called forward–backward soft-input soft-output (SISO) algorithm [5], which is a generalization of the well-known BCJR algorithm [7]. Such a receiver is shown in Fig. 1(b) for the case of SCCC with perfect CSI [the system in Fig. 1(a)], where the notation is similar to [5] (the block denoted SODEM is a soft-output demodulator, i.e., the soft inverse of the modulator). Similarly, the ID network for a PCCC code can be constructed using simple soft inverse blocks, as shown in Fig. 2(b).

In many practical situations, perfect CSI is not available at the receiver. Consequently, an iterative receiver should be able to deal with the unknown, and possibly time varying parameters. The systematic procedure for building ID receivers presented in [5], can be extended to the nonperfect-CSI case as follows: adaptive soft inverse (ASI) blocks—each one corresponding to a subsystem in the original network—are constructed and connected in an intuitive way to form the adaptive ID (AID) network. Several approaches have been suggested in the literature.

In the simplest scenario, the perfect-CSI iterative receiver of Figs. 1(b) and 2(b) are preceded by an external estimator (i.e., a phase-locked loop [PLL] in the case of unknown carrier-phase),
An alternative approach considers both the modulator and the channel as a separate subsystem, and a soft inverse block for this subsystem is derived, namely the adaptive-soft-output demodulator (A-SODEM). In this way, the estimation process is tied with the detection process through the exchange of soft or hard information between the A-SODEM and the rest of the ID network, as shown in Figs. 1(c) and 2(c), for SCCC and PCCC, respectively. This is roughly the approach followed in [9] and [10] for the case of decoding of PCCC’s in a flat fading channel.

Finally, when the inner CC of a SCCC, the modulator, and the channel are treated as a combined subsystem, the ID structure depicted in Fig. 1(d) emerges, where the corresponding soft inverse block is the adaptive SISO (A-SISO) algorithm introduced in [1], [2], [11], that jointly estimates the unknown parameters and provides soft information on the data symbols. An algorithm with a similar structure was developed in [12]. While this approach is straightforward to implement for SCCC, this is not the case for PCCC, since both CC’s are directly involved in the generation of the output symbol. Modified ID structures, based on A-SISO’s can also be built for PCCC’s, as will be described in detail in this work.

In this paper we consider the above approaches to AID for phase tracking in SCCC and PCCC systems. In Section II we develop two basic ASI blocks, namely the A-SISO and the A-SODEM, by appealing to the framework established for the linear estimation problem in [1], [2], [13]. We then apply these algorithms for AID for phase tracking in SCCC (Section III) and PCCC (Section IV). These results demonstrate that the dramatic gains in performance associated with turbo codes relative to traditional convolutional coding are maintained in the presence of severe phase dynamics if and only if a well-designed AID scheme is used.

II. PRACTICAL ASI ALGORITHMS FOR PHASE TRACKING IN TURBO CODES

A. Practical A-SISO Algorithms

Consider an isolated FSM, shown in Fig. 3(a), with input $x_k$, state $s_k$—together constituting the transition $t_k = (s_k, x_k)$— and output $y_k$, where each integer quantity $y_k$ (i.e., $x_k, y_k, s_k$, or $t_k$) is assumed to take values in the set $A_y = \{0, 1, \ldots, N_y - 1\}$. The output $y_k$ is mapped to the complex symbol $\gamma_k$ and transmitted on an AWGN channel, which introduces phase offset $\theta_k$, as well, resulting in the following complex baseband model

$$z_k = \gamma_k e^{j\theta_k} + n_k \tag{1}$$

where the symbols $\gamma_k$ are normalized to unit average energy, $n_k$ is complex circular additive white Gaussian noise (AWGN) with $E\{|n_k|^2\} = N_0/E_s$, and $E_s$ is the symbol energy.

In [1], [13] a slightly different observation model was assumed, where the unknown parameter is linearly dependent with the observation as shown in (2)

$$z_k = \gamma_k g + n_k \tag{2}$$

In particular, under the assumption of the unknown parameter being a deterministic but unknown constant, algorithms were developed for the exact evaluation of the extrinsic information on a generic quantity $u_k$ (i.e., $x_k, y_k, s_k$, or $t_k$) based on the entire observation record, defined by

$$\text{APP}(u_k) \overset{\text{def}}{=} c \sum_{x_0^n} \max_g P(x_0^n, x_0^g | g) \tag{3}$$
where $x_k^0$ denotes the vector $[x_0, \ldots, x_n]$, $x_k^n$ : $u_k$ denotes all input sequences $x_k^n$ consistent with $u_k$, and $\epsilon$ is a normalizing constant. Furthermore, in [1], [2] exact algorithms were developed for the case of the unknown parameter being modeled as a Gauss Markov (GM) random process. As could be expected from the hard-decision literature [14], these exact algorithms have exponential complexity in the sequence length $n$, thus sub-optimal practical algorithms were developed. Among the multitude of algorithms developed in [1], [2], [13], we now describe two basic families of practical algorithms, the basic characteristics of which are:

- Forward/backward recursions on a trellis, similar to the recursions performed in the BCJR [7], or the perfect-CSI SISO [5] algorithms.
- Multiple, per-state parameter estimation, in a way similar to the per-survivor processing (PSP) [14] hard-decision algorithms, both in the forward and in the backward direction. Alternatively, parameter estimation is performed in both directions using a single external delayed decision directed estimator, as in [15]–[17], for hard decision algorithms.
- The existence of a binding term that quantifies the agreement of the forward and backward phase estimates. This term is only present in the nonperfect-CSI case and is a direct consequence of the theoretical framework developed in [2], [11], [13].

To obtain practical A-SISO algorithms for the nonlinear parameter model in (1), we suggest replacing the parameter estimators in the original A-SISO’s by some nonlinear estimator, i.e., in a similar manner suboptimal nonlinear parameter estimators were used in conjunction with PSP in the hard-decision literature [18], [19]. In this paper, a simple first order decision directed phase-locked loop (DD-PLL) is used

$$\hat{\theta}_k = \hat{\theta}_{k-1} + \lambda \Im\{z_k^0 u_k e^{-j \hat{\theta}_{k-1}}\} = f(\hat{\theta}_{k-1}, u_k)$$  \hspace{1cm} (4)$$

where the design parameter $\lambda$ controls the PLL noise equivalent bandwidth [20] (normalized to the symbol time) $\Delta f_{\text{eq}} = \lambda/(4 - 2 \lambda)$, and $\Im\{z_k^0\}$ denotes the imaginary part of $\alpha$. The resulting algorithms, named A-SISO-MULT (i.e., multiple estimators) and A-SISO-SING (i.e., single estimator), are shown in Fig. 4, and are precisely described in Appendices A and B, respectively. In addition, by setting the binding term in (15) equal to zero, two variations of the algorithms described in Appendices A and B emerge, namely A-SISO-MULT-NB (i.e., no binding) and A-SISO-SING-NB, respectively. All of these algorithms are summarized in Table I.

At this point we emphasize that the trellis on which these algorithms operate is not tightly related to the trellis implied by the FSM model. In fact, we can generalize the notion of the state $s_k^0$ and transition $t_k$ to longer sequence portions. As an example, a super-state and super-transition can be defined as $s_k^N = (s_k^N, x_k, x_{k+1}, \ldots, x_{k+N})$ and $t_k^N = (s_k^N, x_k)$ for arbitrary $K$, which is a design parameter that determines the amount of pruning in the forward and backward trees, and eventually, the complexity of the algorithm. This technique has been used extensively in the problem of data detection in the presence of unknown parameters (e.g., [21], [22]).

### B. Practical A-SODEM Algorithms

We now consider the isolated system depicted in Fig. 3(b), where an uncoded sequence is modulated and transmitted through the channel described earlier. We define the A-SODEM as the soft inverse block that provides extrinsic information on $x_k^0$ based on the observation $x_k^n$. Clearly, the system in Fig. 3(b) can be thought of as a special case of the system shown in Fig. 3(a), with an identity FSM having the input and output equal to $x_k^0$ at each time $k$. In this sense, the A-SODEM is an A-SISO corresponding to this identity FSM. We emphasize that the corresponding algorithm trellis can be arbitrary. In particular, defining the super-state $s_k^N = (x_k, x_{k+1}, \ldots, x_{k+N})$, a family of practical algorithms can be derived in a straightforward way having structure similar to the A-SISO’s shown in Fig. 4. We are especially interested in the simplest case of $K = 0$, since this will provide useful links with existing adaptive algorithms. Under this simplifying assumption, the algorithm trellis reduces to a single-state trellis with $N_x$ parallel transitions and the resulting A-SODEM algorithms simplify considerably. The resulting family of A-SODEM’s is named A-SODM-SS (i.e., single state), and is described in Appendix C. One can also envision different variations of the algorithm described in Appendix C. We are especially interested in the three variations described below, which are also summarized in Table I.

- The backward recursion is eliminated, which implies that the binding term in (15) can be set to zero, since there is no backward metric to be bound with the forward metric. This algorithm is named A-SODEM-SS-FW (i.e., forward recursion only).
- The transition metric used in the forward recursion is simplified by eliminating the a priori soft information, i.e., (16) is simplified to $g(x, \theta) = (E_s/N_0)|2 \pi x - m(\theta)||^2$, where all quantities are defined in Appendices A and C.
Furthermore, the backward recursion is eliminated, and
the binding term in (15) is set to zero. Since the soft infor-
mation is not utilized, this algorithm is a nonitera-
tive scheme which needs to be activated only once, before
the iterative decoding process. This algorithm is exactly
the one proposed in [8] and is named EXT (i.e., external
PLL) herein.

- The transition metric used in the forward and backward
recursions is simplified by eliminating the dependence
on the observation, i.e., (16) is simplified to
and the binding term in (15) is set to zero. Under
the first simplification, the forward and backward de-
modulation steps produce the same hard estimate on the
symbol . This, together with the second simplification,
implies that the backward step can be eliminated. This is
roughly the approach followed in [10] for PCCC’s in flat
fading channels and can be summarized as follows for the
phase tracking problem: The incoming soft information
 is thresholded to extract the corresponding hard
decisions . A DD-PLL runs in the forward direction
up to decide the phase estimates. New soft information is
calculated as
where is the phase estimate corresponding to time
, and all quantities are defined in Appendixes A and
C. This algorithm is named A-SODEM-HD (i.e., hard
decisions).

### III. SCCC With Carrier Phase Tracking

As shown in Fig. 1(a), in a SCCC the sequence of source bits
 are partitioned into blocks and convolutionally encoded using
a rate outer CC, producing coded symbols . These sym-
bols are fed to an inner CC of rate through a pseudorandom
symbol interleaver of length . The output symbols are
mapped onto a constellation of size , resulting in an overall
code rate of (bits per channel use). The com-
plex symbols are transmitted through an AWGN channel, re-
sulting in the complex baseband model of (1).

The effectiveness of the adaptive iterative detection algorithm
can be assessed by a number of factors. For example, loss of
lock probability, tracking bandwidth, and BER in the tracking
mode are all relevant performance measures. Initial experiments
suggested that cycle slipping was a major performance limiting
factor. This is because the operating SNR is very low and the
block length (interleaver size) is large. Thus, we consider the
insertion of pilot symbols. In particular, pilot symbols are
inserted in the transmitted sequence for every coded sym-
bols. The pilot insertion loss is accounted for by lowering the
transmitted symbol energy as

\[ E_s = E_b R R_\ell = E_b R_\ell R_\ell \frac{N_d}{N_d + N_t} \log_2 Q \]  

where is the energy per information bit.

#### A. Receivers

The structure of a SCCC is one of a serial concatenation of
two FSM’s through an interleaver and therefore it permits the
iterative receiver shown in Fig. 1(b) for the case of perfect CSI.

Similarly, the block diagram of the iterative receiver that uti-
lizes an A-SODEM and inner and outer nonadaptive SISO’s is
shown in Fig. 1(c), while the A-SISO based receiver is shown
in Fig. 1(d).

#### B. Numerical Results

The SCCC system simulated in this section consists of an
outer 4-state, rate CC, connected through a length symbol pseudo-random interleaver to an inner 4-state,
rate CC. The corresponding generator matrices are given by

\[
G_{\text{outer}}(D) = \begin{bmatrix}
1 & 0 \\
\frac{1+D^2}{1+D+D^2} & 1
\end{bmatrix},
\]

\[
G_{\text{inner}}(D) = \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}.
\]

The output symbols are mapped to an 8-ary Phase Shift Keying
(8PSK) constellation with Gray encoding, resulting in an overall
code rate \( R = \frac{1}{2} \times 2/3 \times \log_2 8 = 1 \). The phase process is
generated as a random walk as in [23]

\[ \theta_k = \theta_{k-1} + \psi_k \]

where is a Gaussian increment of zero mean and variance .

The algorithm trellises are identical to the corresponding FSM
was optimized for each to
of 0.04 dB. The PSP-based receiver oper-
are gradually increased until the
) a clear advantage of the A-SISO-MULT
is
at 4 dB.
, resulting in
, better tracking is possible,
for
suggests the ability to track larger phase dynamics.
, 128-state CC [24] p. 493
, with only a loss in the effective
values (i.e., 2 dB),
for SCCC with phase dynamics and various
values are presented. Receivers employing external PLL, as well
as inner adaptive SISO’s are considered.

In Fig. 5 the BER is plotted versus the loop bandwidth $B_{eq}$ for
the case of the true phase process being static ($\sigma_\beta = 0$). A large
value of $B_{eq}$ suggests the ability to track larger phase dynamics. The comparison of EXT and A-SISO-MULT curves leads to dif-
ferent conclusions depending on the bandwidth range: In the low
loop-bandwidth range ($B_{eq} \leq 10^{-3}$) the two receivers perform
almost identically, approaching the perfect-CSI performance,
thus the EXT receiver suffices. For medium and high loop-band-
width ($B_{eq} > 10^{-5}$) a clear advantage of the A-SISO-MULT
can be observed over the EXT receiver. In particular, the sim-
ulations show that with the proposed algorithm the PLL band-
width can be increased two to three times.

Regarding the comparison between pilot-aided and non-
pilot-aided, the basic trade-off is controlled by the parameter
$N_f$ (for fixed $N_d$): by increasing $N_f$, better tracking is possible,
while the symbol energy $E_s$ is reduced as reflected in (5). In
the one extreme, no pilots are introduced ($N_f = 0$), resulting in
high probability of cycle slipping at moderate phase dynamics. In
the other extreme ($R_t \ll 1$), the pilot insertion loss nullifies
any performance gain due to the improved phase estimate. Two practical cases are shown in Fig. 5: nonpilot-aided and
($N_f, N_d = (32, 256)$) pilot-aided transmission. At low $E_b/N_0$
(i.e., 1 dB) the nonpilot-aided system is superior, since the
pilot insertion loss is $-10 \log_{10} R_t = 0.51$ dB, reducing the effective
$E_s/N_0$ to 1$ - 0.51 = 0.49$ dB, which results in poor
performance even in the coherent case. At medium $E_b/N_0$ (i.e.,
1.5 dB) the trade-off is reversed, generating a two- to three-fold
advantage of the pilot-aided system over the nonpilot-aided
one in terms of $B_{eq}$. This behavior is attributed to the fact that
the former system is able to maintain phase lock for wider
loop bandwidths. Finally, at large $E_b/N_0$ values (i.e., 2 dB),
the superiority of the pilot-aided system is even more evident,
giving rise to as much as five to seven times increase in $B_{eq}$
and achieving even lower BER. The above comparisons raise
the issue of proper selection of the system parameter ($N_f, N_d$).
Our design procedure is initiated by setting a target BER and
$B_{eq}$ region. A search procedure is then followed, in the process
of which, $E_b/N_0$ and $N_f$ are gradually increased until the target
(BER,$B_{eq}$) pair is reached. Regarding the selection of
$N_d$, it should be smaller than the average time-to-slip or else
the performance will be dominated by cycle slips.

Fig. 6 shows a comparison of the SCCC system with the maxi-
mal free distance, rate 1/2, 128-state CC [24] p. 493, in the
more realistic scenario that includes phase dynamics. The CC
output is mapped on a quadrature phase-shift keying (QPSK) al-
phabet resulting in a rate $R = 1$ (bits per channel use) code (no
pilot symbols are used). MLSD is performed using the Viterbi
algorithm (VA) in the coherent case, while two adaptive receiver
structures are considered. The first is the conventional adap-
tive-MLSD (CA-MLSD) receiver of [17], consisting of a single
DD-PLL driven by delayed tentative decisions from the VA, and
the second is a PSP-based [14] receiver consisting of a VA with
128 DD-PLL’s driven with zero-delay decisions. Simulations
were run for $\sigma_\beta = 2^3$ and $B_{eq}$ was optimized for each $E_b/N_0$
value. Examining the CC performance curves, the following ob-
servations can be made. With perfect CSI, a BER of $10^{-5}$ is
achieved at $E_b/N_0 = 3.75$ dB. The PSP-based receiver operates
at this BER with a loss of 0.4 dB, while the the CA-MLSD
receiver performs poorly resulting in a BER of $10^{-2}$ at 4 dB.

The design procedure outlined in the previous paragraphs
was followed for the selection of $N_f$ for the SCCC case. Simulation trials not shown here suggested that a reasonable
pair is \((N_1, N_2) = (16, 256)\) for a target BER of \(10^{-5}\) and a phase process generated as in (7) with \(\sigma_0 = 2^9\). Observing the SCCC curves in Fig. 6 we conclude that A-SISO-MULT and A-SISO-SING \((d = 0)\) perform identically \((0.6 \text{ dB} \text{ away from the coherent case at a BER of } 10^{-5})\). This may be attributed to the fact that the former corresponds to an FSM of only 4 states. Therefore, there is no notable gain by using four PLL’s instead of one PLL. This is to be contrasted with the 128-state CC case, where a large difference between the CA-MLSD and PSP-based decoders is observed. A small degradation of \(0.2 \text{ dB} \text{ at a BER of } 10^{-5}\) is observed when binding is dropped in the A-SISO-SING-NB receiver. The A-SODEM-SS receiver has very similar performance with the A-SISO-MULT and A-SISO-SING-NB receivers, even though it corresponds to a single-state FSM, while the simplified version A-SODEM-SS-FW, results in a performance loss of \(1 \text{ dB} \text{ at a BER of } 10^{-5}\) compared to the best adaptive receivers considered.

Regarding the performance of the proposed adaptive schemes, it can be noted that since the pilot insertion loss is \(-10 \log_{10} R_t = 0.27 \text{ dB}\), the actual loss due to the unknown phase is only \(0.33 \text{ dB}\) for the best adaptive scheme, i.e., A-SISO-MULT/SING, at a BER of \(10^{-5}\). This means that if the state space is increased by using super-state-based ASI’s), but the pilot symbol structure remains fixed, the expected performance gain is at most \(0.33 \text{ dB}\). On the other hand, by using super-state-based ASI’s, it may be possible to achieve the same performance with less overhead due to pilot symbols, with a potential gain of \(0.6 \text{ dB}\).

The comparison of the CC and SCCC curves clearly illustrates the importance of adaptive iterative detection. Under perfect CSI, the SCCC performs with a \(2.7-\text{dB} \text{ gain over the standard CC at BER of } 10^{-5}\). This gain vanishes when a PSP based MLSD receiver is used to decode CC and the EXT receiver is used for SCCC. By utilizing the more advanced A-SISO’S or A-SODEM’s proposed herein, together with pilot symbols, the corresponding gain is \(2.6 \text{ dB at a BER of } 10^{-5}\).

**References**

**IV. PCCC WITH CARRIER PHASE TRACKING**

In a PCCC [3], a length \(N\) block of the original sequence \(x_k\) is encoded by a rate \(R_2\) CC, while an interleaved version of the input sequence is encoded by a second CC of rate \(R_2\), giving rise to the coded symbols \(y_{k}^{(1)}\) and \(y_{k}^{(2)}\), respectively. The output symbols \(y_{k}^{(1)}\) and \(y_{k}^{(2)}\) are then mapped—after possible puncturing—to the symbols \(q_{k}\) and transmitted over an AWGN channel which introduces phase uncertainty, modeled exactly as in the case of SCCC’S. The observation equation is written as

\[
z_{k} = m_{k}\left(y_{k}^{(1)}, y_{k}^{(2)}\right)e^{i\theta_{k}} + n_{k} = q_{k}e^{i\theta_{k}} + n_{k} \tag{8}
\]

where the time-varying mapping \(q_{k} = m_{k}(y_{k}^{(1)}, y_{k}^{(2)})\) is explicitly shown. Pilot symbols are inserted in the transmitted sequence in the same manner described in the previous section.

For concreteness, we consider the case where \(y_{k}^{(1)} = (x_{k}, c_{k}^{(1)})\) consists of a systematic and a coded bit \((R_1 = 1/2)\), \(y_{k}^{(2)} = c_{k}^{(2)}\) is a binary coded bit \((R_2 = 1)\) and \(q_{k}\) belongs to a QPSK signal constellation. This signaling format, which was the basis for the original turbo code, can be achieved by transmitting the systematic bit \(x_{k}\), together with the coded bits \(c_{k}^{(1)}\) or \(c_{k}^{(2)}\) after alternate puncturing.

\[
q_{k} = m_{k}\left(y_{k}^{(1)}, y_{k}^{(2)}\right) = \begin{cases} \text{QPSK}(x_{k}, c_{k}^{(1)}), & k \text{ even} \\ \text{QPSK}(x_{k}, c_{k}^{(2)}), & k \text{ odd} \end{cases} \tag{9}
\]

where QPSK(*) maps the bits to the two-dimensional QPSK signal constellation (e.g., Gray mapping is used in the numerical results presented herein).

**A. Receivers**

Since PCCC’S can be modeled as parallel concatenated FMS’s, the iterative decoder shown in Fig. 2(b) can be applied when perfect CSI is available. The A-SODEM-based receiver is shown in Fig. 2(c). All A-SODEM variants discussed earlier (i.e., A-SODEM-SS, S-SODEM-SS-FW, and EXT) can be used herein unchanged. However, for the A-SISO-based receiver, in contrast to the serially concatenated examples considered in Section III, the PCCC has the property that the outputs of both FMS’s are directly affected by the channel. Furthermore, the outputs of the constituent FMS’S are coupled via the nonlinear mapping (8). This makes the substitution of the perfect-CSI SISO by an A-SISO insufficient for performing adaptive iterative detection in this case. Thus, adaptive iterative detection for this PCCC application requires a method for evaluating transition metrics and updating phase estimates for each A-SISO. In the following we discuss the options for doing so and demonstrate one specific approach.

**1) Metric Evaluation:** Metric evaluation in A-SISO1 can be performed by treating the output symbols corresponding to CC2 as nuisance parameters and averaging over them. For example, the transition metric used in (14) for the recursion in SISO1 is evaluated as

\[
\gamma_k(t, \theta) = \text{SL}_{k}(\text{in}(t)) - \log \left[ \sum_{y^{(2)}} P(y^{(2)}) \right] \\
\times \exp \left\{ -\frac{E_s}{N_0}\left[ z_{k} - m_{k}(\text{out}(t), y^{(2)})e^{i\theta} \right]^{2} \right\} \\
= \text{SL}_{k}(\text{in}(t)) - \min_{y^{(2)}} \left[ -\log P(y^{(2)}) \right] \\
+ \frac{E_s}{N_0}\left[ z_{k} - m_{k}(\text{out}(t), y^{(2)})e^{i\theta} \right]^{2} \tag{10}
\]

where all quantities are defined as in Appendix A. A reasonable choice for the probability \(P(y^{(2)})\) is to use the most recent soft-information produced by A-SISO2. This is identical to the operation of soft mapper (SOMAP) [5] in the case of perfect CSI. The only difference is that the demodulator and the SOMAP are now integrated with the A-SISO1, since a phase estimate is required for this operation. This solution is both simple to implement, and compatible with the notion that SISO blocks exchange information only in the form of soft metrics. A similar procedure can be followed for the evaluation of the transition metrics of A-SISO2.
2) Parameter Estimate Update: Several options are considered for updating the phase estimate in A-SISO1.

Starting from the simplest solution, the channel update in A-SISO1 is only performed for those time instants $k$, for which the symbol $q_k$ is only a function of $y_k^{(1)}$ ($k$ is even). The resulting updates for this punctured DD-PLL (P-DD-PLL) become

$$
\hat{\theta}_k = \begin{cases} 
\hat{\theta}_{k-1} + \lambda \left\{ z_k \text{QPSK} \left( x_k c_k^{(1)} \right)^* e^{-j\hat{\theta}_{k-1}} \right\}, & k \text{ even} \\
\hat{\theta}_{k-1} + \lambda \left\{ z_k \text{QPSK} \left( x_k c_k^{(2)} \right)^* e^{-j\hat{\theta}_{k-1}} \right\}, & k \text{ odd}
\end{cases},
$$

(11)

This approach can be used both, in multiple- and single-PLL A-SISO’s. The immediate consequence of this sort of channel update is a loss of the full tracking ability of the estimator (i.e., the effective loop bandwidth is halved). In addition, such an approach is not always applicable, since the mapping $m_k(\cdot)$ may always be an explicit function of the symbol $c_k^{(1)}$ as well, as in the case of nonpunctured codes. Furthermore, this method cannot be used for A-SISO2, because the systematic bit $x_k$ has been punctured in the lower constituent code. As a result, the update equation corresponding to (11) for A-SISO2 degenerates to

$$
\hat{\theta}_k = \hat{\theta}_{k-1},
$$

which means no update at all.

In order for the DD-PLL (or DD-PLL’s) to be updated for every time instant $k$, an estimate $\hat{\theta}_k$, of $\theta_k$, is required when $k$ is odd. The later can be determined by thresholding the most recent soft information on available at the output of A-SISO2. The updates for this hard decision directed PLL (H-DD-PLL) are given by

$$
\hat{\theta}_k = \begin{cases} 
\hat{\theta}_{k-1} + \lambda \left\{ z_k \text{QPSK} \left( x_k c_k^{(1)} \right)^* e^{-j\hat{\theta}_{k-1}} \right\}, & k \text{ even} \\
\hat{\theta}_{k-1} + \lambda \left\{ z_k \text{QPSK} \left( x_k c_k^{(2)} \right)^* e^{-j\hat{\theta}_{k-1}} \right\}, & k \text{ odd}
\end{cases},
$$

(12a)

Similarly, the updates for the H-DD-PLL in A-SISO2 are given by

$$
\hat{\theta}_k = \begin{cases} 
\hat{\theta}_{k-1} + \lambda \left\{ z_k \text{QPSK} \left( \tilde{x}_k c_k^{(1)} \right)^* e^{-j\hat{\theta}_{k-1}} \right\}, & k \text{ even} \\
\hat{\theta}_{k-1} + \lambda \left\{ z_k \text{QPSK} \left( \tilde{x}_k c_k^{(2)} \right)^* e^{-j\hat{\theta}_{k-1}} \right\}, & k \text{ odd}
\end{cases},
$$

(12b)

where $\tilde{x}_k$ and $c_k^{(1)}$ are hard decisions derived from the corresponding soft-decisions from A-SISO1.

One can also design a mixed-mode PLL. Such a PLL—for A-SISO1—operates in a decision directed mode in terms of the symbol $y_k^{(1)} = (x_k c_k^{(1)})$, while it effectively averages out the symbol $c_k^{(2)}$. We do not pursue this design further and direct the reader to [25], where a simple PLL structure that operates by averaging equiprobable binary symbols is discussed.

In the following, transition metrics are evaluated by averaging out the symbols corresponding to the other FSM as described in (10). In addition, a hybrid approach for phase tracking is used. Specifically, A-SISO1 is run with the P-DD-PLL of (11) on the initial iteration, and switches to the H-DD-PLL of (12a) in the subsequent iterations, while A-SISO2 runs with a H-DD-PLL. The rational behind this hybrid bootstrapping procedure is that in the first iteration, there are no soft (or hard) decisions available for the symbol $y_k^{(2)}$, which necessitates the use of a P-DD-PLL in A-SISO1. After the activation of A-SISO1, soft—and thus hard—decisions are available for $y_k^{(1)}$, which enables the use of H-DD-PLL in A-SISO2. The activation schedule for the iterative detector shown in Fig. 2(d), is described as follows: A-SISO1 (with P-DD-PLL) $\rightarrow$ A-SISO2 (with H-DD-PLL) $\rightarrow$ A-SISO1 (with H-DD-PLL), etc.

B. Numerical Results

An overall rate $R = 1$ code is considered in this section, constructed by concatenating two identical 4-state CC’s, and using a size $N = 16,384$ pseudorandom interleaver. Both the systematic and the encoded bits are output from the first code, while only the encoded bit is output from the second. The corresponding generator matrices are given by

$$
G_1(D) = \begin{bmatrix} 1 & 1+D^2 \end{bmatrix} \quad G_2(D) = \frac{1+D^2}{1+D+D^2}
$$

(13)

The output symbol is formed exactly as described in (9). Furthermore, QPSK pilot symbols with $(N_p, N_d) = (16,256)$ are inserted. In Fig. 7, performance curves similar to those of Fig. 6 are presented. The conclusions are similar to the SCCC case, with the only difference being the slight degradation of the A-SISO-SING and A-SISO-SING-NB algorithms over the A-SISO-MULT receivers. In addition, the A-SODEM-SS-FW performance is very close to the performance of the A-SISO-based receivers, and the A-SODEM-SS algorithm results in slightly better performance compared to A-SISO’s. The later result might seem counterintuitive, however we emphasize that the A-SODEM and the A-SISO are used differently, so it is not the case of one replacing the other. Specifically, for the case using the A-SODEM, there is one A-SODEM and two nonadaptive SISO’s, while for the case using the A-SISO’s, there is no SODEM or A-SODEM present.
Furthermore, iterative detection is an approximation to the optimal solution, and the A-SISO and A-SODEM blocks are approximations to the exact ASI algorithms developed in [1], [2]. In this sense, there is no accurate way to predict the relative performance of these algorithms, other than simulation. An intuitive explanation might be that, unlike SCCC’s, the A-SISO based algorithms require several modifications (including averaging over the unknown symbols of the other FSM) in order for the overall adaptive receiver to be operational, while the A-SODEM based receiver is a very simple modification of the perfect-CSI iterative receiver.

It is observed that, as in the case of perfect CSI, the quantitative performance achieved using the SCCC and PCCC systems is very similar. Finally, simulations for the case of static phase revealed comparable performance with that shown in Fig. 5; these results are not presented for brevity.

V. CONCLUSION

Iterative detection, based on ASI algorithms (i.e., A-SISO and A-SODEM) was presented in this paper. For the detection of turbo codes (i.e., SCCC’s and PCCC’s) and phase tracking for the practical scenarios examined, it was shown that pilot-symbol-assisted adaptive iterative detection is effective for maintaining the near-Shannon-limit performance previously demonstrated for known phase systems. Incorporating the estimation process into the ID process gives advantages similar to those documented in adaptive hard decision literature, where joint estimation and data detection is considered for an isolated system (e.g., PSP-based phase tracking yields a factor of 2–3 in loop bandwidth extension for trellis codes)

There are several directions remaining for future research. The effect of the code and signal selection on the performance was not investigated. The fact that the SCCC is mapped onto an 8PSK constellation presents an additional impediment for the phase estimator. It may be possible to construct a more efficient SCCC using a QPSK constellation (e.g., by puncturing the outer and/or inner code). In fact, for a channel utilization of 1 bit per channel use at low $E_b/N_0$, a QPSK constellation is adequate for achieving capacity [26]. Also, potential further improvement for the SCCC may be achieved by the development of rotationally invariant—possibly multidimensional—inner codes. The use of such codes may alleviate the detrimental effects of cycle slipping, potentially enabling even wider loop bandwidths.

The presentation of the last application, namely decoding of PCCC’s with phase tracking, revealed that the concept of adaptive iterative detection is broader than the concept of ASI algorithms (A-SISO’s or A-SODEM’s). Although practical receivers were proposed based on the latter structures, the development of a general framework for adaptive iterative detection on arbitrary networks based on the theoretical framework developed in [6] is an area for future research.

APPENDIX

A. Multiple-Estimator A-SISO Algorithm (A-SISO-MULT)

In this Appendix, the multiple-estimator A-SISO algorithm is described. The cumulative forward and backward metrics associated with state $s_k = s$ are denoted by $\alpha_k(s)$ and $\beta_k(s)$, respectively. Similarly, the forward and backward phase estimates associated with state $s_k = s$ are denoted by $\hat{\theta}_k(s)$ and $\tilde{\theta}_k(s)$, respectively. The quantities $\ln(t)$, $\ln(t)$, $\ln(t)$, and $\alpha(t)$ denote the input, initial state, final state, and output, associated with transition $t$, respectively. In addition, the quantity $\text{sym}(t) = m(\alpha(t))$ denotes the constellation point associated with transition $t$, where $m(\cdot)$ is the memoryless mapping from the output of the FSM to the constellation point. The transition metric associated with transition $t_k = t$ and phase estimate $\theta$ is defined by (see [1] for details)

$$\gamma_k(t, \theta) \overset{\text{def}}{=} \text{SL}_k(\ln(t)) + \frac{E_b}{N_0} |z_k - \text{sym}(t)e^{\theta}|^2$$

(14)

where $\text{SL}_k(x)$ is the available soft input on the input symbol $x_k = x$. In general, $\text{SL}_k(u)$ is the available soft input, and $\text{SO}_k(u)$ is the calculated soft output, respectively, for the generic quantity $u_k = u$ (where $u_k$ can be any of $x_k$, $y_k$, $s_k$, or $t_k$). All operations are performed in the log domain with the $\min^*$ operator defined by $\min^*(x, y) \overset{\text{def}}{=} \min(x, y) - \log(1 + \exp(-|x - y|))$. Parameter update is performed by using the DD-PLL in (4), which is summarized in the function $f(\cdot, \cdot)$. The binding term is defined as

$$b(\theta^f, \theta^b) \overset{\text{def}}{=} \frac{1 - \lambda}{\lambda(2 - \lambda)} \frac{E_b}{N_0} |e^{\theta^f} - e^{\theta^b}|^2$$

(15)

where $\lambda$ is the PLL parameter in (4). We note that this last equation is an approximation obtained intuitively by the framework in [1]. The above equation provides additional insight on the role of the binding term: If the forward and backward channel estimates corresponding to a particular sequence are not consistent, a penalty is paid by means of increasing the sequence metric. Furthermore, this penalty is amplified when tracking slowly changing parameters ($\lambda$ close to 0).

0) Initialization: $\forall s = 0, \ldots, N_s - 1$

$$\alpha_0(s) = \beta_{n+1}(s) = 0$$

(or according to the available information on the trellis boundaries)

$$\theta^f_{n+1}(s) = \text{initial forward phase estimate}$$

$$\theta^b_{n+1}(s) = \text{initial backward phase estimate}$$

1) Forward recursion: $\forall k = 0, \ldots, n$

$$\forall s = 0, \ldots, N_s - 1$$

Metric update:

$$\alpha_{t+1}(s) = \min^*_{\theta_k}(\alpha_k(\text{sym}(t)) + \gamma_k(t, \theta_{k-1}(\text{sym}(t))))$$

Parameter update:

$$t' = \arg\min_{\theta_k}[\alpha_k(\text{sym}(t)) + \gamma_k(t, \theta_{k-1}(\text{sym}(t)))$$

$$\theta^f_k(s) = f(\theta_{k-1}(\text{sym}(t')), \text{sym}(t'))$$

2) Backward recursion: $\forall k = n, \ldots, 0$

$$\forall s = 0, \ldots, N_s - 1$$

Metric update:

$$\beta_k(s) = \min^*_{\theta_k}[\beta_{k+1}(\text{sym}(t)) + \gamma_k(t, \theta^f_{k+1}(\text{sym}(t)))]$$

Parameter update:

$$t' = \arg\min_{\theta_k}[\beta_{k+1}(\text{sym}(t)) + \gamma_k(t, \theta^f_{k+1}(\text{sym}(t)))$$

$$\theta^b_k(s) = f(\theta^f_{k+1}(\text{sym}(t')), \text{sym}(t'))$$
3) Completion stage: \( \forall k = 0, \ldots, n \)
\[ \forall u = 0, \ldots, N_u - 1 \]
\[ \text{SO}_k(u) = \infty \]
\[ \forall t : u \]

(*) Extend channel estimate:
\[ \hat{\theta}^f = f(\hat{\theta}_{k-1}(\text{sym}(t)), \text{sym}(t)) \]
Evaluate binding term:
\[ b = \mathcal{U}(\hat{\theta}^f, \hat{\theta}_{k+1}(\text{ns}(t))) \]
\[ \text{SO}_k(u) = \min \{ \text{SO}_k(u), \text{SO}_{k+1}(u) \} + \gamma_k(t, \hat{\theta}_{k-1} + \beta_{k+1}(\text{ns}(t)) + b \} \]
Output extrinsic information:
\[ \text{SO}_k(u) = \text{SO}_k(u) - \text{SI}_k(u) \]

We note that step (*) can be omitted and \( \hat{\theta}^f = \hat{\theta}_{k-1}(\text{is}(t)) \) can be used in evaluating the binding term.

B. Single-Estimator A-SISO Algorithm (A-SISO-SING)

The single-estimator A-SISO algorithm is now described. In this case, a single external forward \((\hat{\theta}_k)\) and backward \((\hat{\theta}_k)\) phase estimate is stored and updated for each time \(k\). The quantity \(\text{TSO}_k(u | x_k^2)\) is a tentative soft output of the symbol \(y\) based on the observation \(x_k^2\) and is evaluated as follows: at time \(k\) a backward nonadaptive sum-product recursion is initiated, in the same way done in the BCJR (or SISO) algorithm. After \(d\) backward steps, the forward, backward and transition metrics are combined and marginalized to produce the tentative soft output. The quantity \(\text{TSO}_k(u | x_k^2)\) is defined and evaluated similarly.

0) Initialization:
\[ \forall s = 0, \ldots, N_s - 1 \]
\[ \alpha_0(s) = \beta_{k+1}(s) = 0 \] (or according to the available information on the trellis boundaries)
\[ \hat{\theta}_{k+1} = \text{initial forward phase estimate} \]
\[ \hat{\theta}_{k+1} = \text{initial backward phase estimate} \]
1) Forward recursion: \( \forall k = 0, \ldots, n \)
\[ \forall s = 0, \ldots, N_s - 1 \]
Metric update:
\[ \alpha_{k+1}(s) = \min \{ \alpha_k(s), \beta_{k+1}(s) + \gamma_k(t, \hat{\theta}_k) \} \]
Parameter update:
\[ d \] forward sum-product steps to obtain
\[ \text{TSO}_k(u | x_k^2) \]
tentative decision:
\[ q^f = m(\text{argmin}_{u} \{ \text{SI}_k(u, d) + \text{TSO}_k(u, d) | x_k^2 \}) \]
\[ \hat{\theta}_{k-1} = f(\hat{\theta}_{k-1}, q^f) \]
2) Backward recursion: \( \forall k = n, \ldots, 0 \)
\[ \forall s = 0, \ldots, N_s - 1 \]
Metric update:
\[ \beta_k(s) = \min \{ \beta_k + \gamma_k(t, \hat{\theta}_k) \} \]
Parameter update:
\[ d \] forward sum-product steps to obtain
\[ \text{TSO}_k(u | x_k^2) \]
tentative decision:
\[ q^b = m(\text{argmin}_{u} \{ \text{SI}_k(u, d) + \text{TSO}_k(u, d) | x_k^2 \}) \]
\[ \hat{\theta}_{k+d+1} = f(\hat{\theta}_{k+d+1}, q^b) \]
3) Completion stage: \( \forall k = 0, \ldots, n \)
\[ \forall u = 0, \ldots, N_u - 1 \]
\[ \text{SO}_k(u) = \infty \]

\[ \forall t : u \]

(*) Extend channel estimate:
\[ \hat{\theta}^f = f(\hat{\theta}_{k-1}(\text{sym}(t)), \text{sym}(t)) \]
Evaluate binding term:
\[ b = \mathcal{U}(\hat{\theta}^f, \hat{\theta}_{k+1}(\text{ns}(t))) \]
\[ \text{SO}_k(u) = \min \{ \text{SO}_k(u), \text{SO}_{k+1}(u) \} + \gamma_k(t, \hat{\theta}_{k-1} + \beta_{k+1}(\text{ns}(t)) + b \} \]
Output extrinsic information:
\[ \text{SO}_k(u) = \text{SO}_k(u) - \text{SI}_k(u) \]

C. Single-State A-SODEM Algorithm (A-SODEM-SS)

For this trivial FSM, it is true that \( x_k = y_k = t_k \). The transition metric now can be written as
\[ \gamma_k(t, \theta) = \gamma_k(x, \theta) = \text{SI}_k(x) + \frac{E_b}{N_0} | z_k - m(x)e^{j\theta}|^2 \]
(16)
The resulting algorithm is as follows

0) Initialization:
\[ \hat{\theta}_{k+1} = \text{initial forward phase estimate} \]
\[ \hat{\theta}_{k+1} = \text{initial backward phase estimate} \]
1) Forward recursion: \( \forall k = 0, \ldots, n \)
Demodulate:
\[ x' = \text{argmin}_{\theta} \gamma_k(x, \hat{\theta}_{k+1}) \]
Parameter update:
\[ \hat{\theta}^f = f(\hat{\theta}_{k+1}, m(x')) \]
2) Backward recursion: \( \forall k = n, \ldots, 0 \)
Demodulate:
\[ x' = \text{argmin}_{\theta} \gamma_k(x, \hat{\theta}_{k+1}) \]
Parameter update:
\[ \hat{\theta}^b = f(\hat{\theta}_{k+1}, m(x')) \]
3) Completion stage: \( \forall k = 0, \ldots, n \)
\[ \forall x = 0, \ldots, N_x - 1 \]

(*) Extend channel estimate:
\[ \hat{\theta}^f = f(\hat{\theta}_{k-1}, m(x)) \]
Output extrinsic information:
\[ \text{SO}_k(x) = \frac{(E_b)}{(N_0)} | z_k - m(x)e^{j\theta}|^2 + b(\hat{\theta}^f, \hat{\theta}_{k+1}) \]

REFERENCES


