SYSTEM DESIGN FOR FEC IN AERONAUTICAL TELEMETRY

Erik Perrins
Department of Electrical Engineering & Computer Science
University of Kansas
Lawrence, KS 66045
esp@ieee.org

ABSTRACT

This paper contains a description of two types of forward error correction (FEC) codes for shaped offset quadrature phase shift keying, telemetry group version (SOQPSK-TG). The FEC codes are a low-density parity check (LDPC) code and a serially concatenated convolutional code (SCCC).

The contributions of this paper are on the system-design level. One major contribution is to design a SCCC code word format that is as compatible as possible with the LDPC code word, which simplifies other aspects of the system design. Another major contribution is to show exactly how demodulators and decoders can be decoupled from each other at the receiver. This simplifies the demodulation process because receiver synchronization is no longer intertwined with FEC decoding. Furthermore, this enables a mix-and-match design, where demods can be chosen based on their performance and complexity tradeoffs. In fact, for the first time, we show how *symbol-by-symbol* demods can be used with all FEC coding/decoding options, and we also show that these demods have very attractive BER performance given their simplicity.

TRANSMITTER MODEL

FEC Encoders

The transmitter model is shown in Figure 1. The information word (sequence) is denoted as $\mathbf{u} \triangleq \{u_i\}_{i=0}^{K-1}$, where K is the number of bits contained in the information word and each bit has a duration of T_b seconds. The FEC encoder accepts $\mathbf{u}$ as its input and returns the code word (sequence) $\mathbf{c} \triangleq \{c_i\}_{i=0}^{N-1}$ as its output, where N is the number of symbols contained in the code word and each symbol has a duration of T_s seconds. The FEC encoder has a rate $R \triangleq K/N$ and we have the relationship $T_s = RT_b$.

Based on analysis of link budget and throughput requirements, the iNET Communication Link Standards Working Group (CLSWG) identified a coding rate of $R = 2/3$ and an information word length of $K = 4096$ bits (code word length of $N = 6144$ symbols) as attractive design choices [1]. Two FEC options were identified by the CLSWG: LDPC and SCCC. Furthermore, based on its successful deployment in serial streaming telemetry (SST) links, the CLSWG identified SOQPSK-TG as the initial option for the physical-layer waveform. Figure 1 reflects the SOQPSK-based transmitter design, where the code word $\mathbf{c}$ is fed to the SOQPSK modulator which produces the transmitted signal $s(t; \mathbf{c})$. The two FEC options and the SOQPSK-TG waveform are now specified in greater detail.

The LDPC code is the $R = 2/3$, $K = 4096$ ($N = 6144$) code developed at NASA's Jet Propulsion Laboratory (JPL) [2, 3]. Because the full specification of this code is found in [3], we give only cursory details here. This is a quasi-cyclic code in the family known as "Accumulate Repeat-4 Jagged Accumulate" (AR4JA) codes. The generator matrix for this code is expressed in *systematic form* as $G = [I_{4096} \ W]$, where I_N denotes the $N \times N$ identity matrix. Because the generator matrix is in systematic form, the LDPC code words have the format shown in Figure 2, where the information word $\mathbf{u}$ occupies the first 4096 positions in $\mathbf{c}$, and the parity symbols occupy the last 2048 positions in $\mathbf{c}$. The full specification of the 4096×2048 matrix W is found in [3]; suffice it to say that it is a dense matrix of 256×256 block circulants that permits a simplified hardware implementation. This code has a native rate of $4/7$ and the desired rate of $2/3$ is achieved by puncturing (deleting) the last 1024 columns of the original generator matrix in [3]. Thus, the sparse parity-check matrix H used in the decoding algorithm has dimensions 3072×7168.

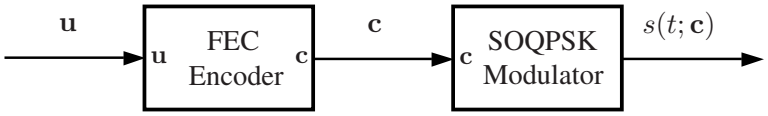


Figure 1: Transmitter Model.

Figure 2: Systematic code word format.

LDPC Encoder

The LDPC encoder is shown in Figure 3, and is basically a trivial repetition of the generic FEC encoder in Figure 1. In order to create LDPC code words, the encoder performs the straightforward operation $\mathbf{c} = \mathbf{u}G = [\mathbf{u} \;\; \mathbf{u}W]$, where the arithmetic is performed modulo-2. When transmitting an LDPC code word, the SOQPSK modulator is operated *without* differential encoding and is viewed as being separate and distinct from the code, as we explain further in the section on decoding algorithms.

SCCC Encoder

The SCCC scheme for aeronautical telemetry was originally developed at the University of Kansas [4, 5]. The description in this paper contains certain design refinements to the basic SCCC scheme, which have the goal of maximizing the system-level compatibility between the LDPC and SCCC schemes. Unlike the LDPC encoder, which is simply a generator matrix, the SCCC encoder consists of two *constituent encoders*, separated by an interleaver, as shown in Figure 4. Because each constituent encoder is an encoder in its own right, each has its own internal input $\mathbf{u}$ and the output $\mathbf{c}$. These modules are connected together to form the overall SCCC input $\mathbf{u}$ and the output $\mathbf{c}$, as detailed below.

CC Encoder. The first of these submodules is a convolutional encoder that contains the $R = 1/2$, 4-state, *systematic feedback* (FB) convolutional code (CC) with a generator given by (7,5) in the octal notation, and in polynomial form is given by [6]

$$\mathbf{G}(D) = \begin{bmatrix} 1 & \dfrac{1+D^2}{1+D+D^2} \end{bmatrix}.$$

A block diagram of the CC encoder is shown in Figure 5. It is depicted as a Direct Form II linear time-invariant (LTI) system [7] with one delay element labeled as least significant bit (LSB) and the other as most significant bit (MSB). The arithmetic is performed modulo-2. The encoder has one output stream containing the information bits (without alteration) and one output stream containing the parity symbols. The two streams are combined by means of a commutator, which results in a final output stream with two symbols for every input bit.

The encoder is initialized to the all-zeros state prior to encoding the information word. After the final information bit, u_{4095}, is clocked into the encoder, two additional *termination symbols* are appended to the input stream, t_{4096} and t_{4097}. The purpose of the termination symbols is to return the encoder to the all-zeros state. Trellis termination

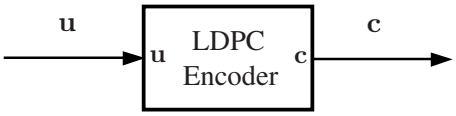


Figure 3: Block diagram of the LDPC encoder.

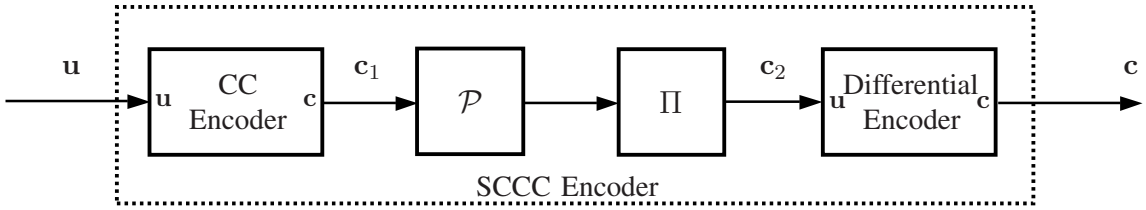
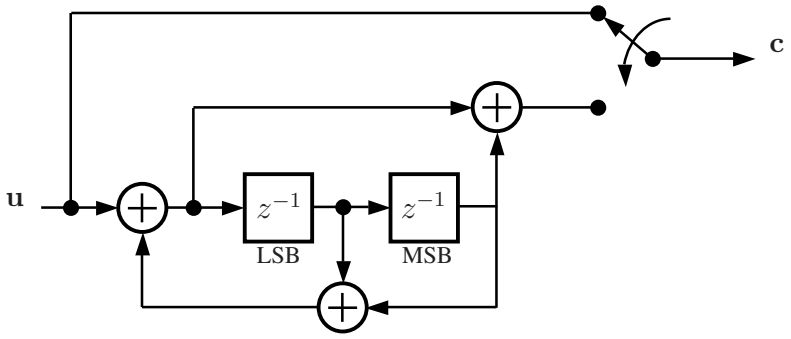


Figure 4: Block diagram of the SCCC encoder.



State (MSB|LSB)	t_{4096}	t_{4097}
00	0	0
01	1	1
10	1	0
11	0	1

Figure 5: On the left is the block diagram of the *R* = 1/2, 4-state, (7,5) systematic feedback (FB) CC encoder. On the right is a table that specifies the termination symbols for the CC encoder.

is beneficial for the receiver, because the decoding algorithm can begin and end its processing at a known state in the trellis. This ensures that the information bits at the beginning and ending edges are decoded as reliably as the ones in the middle. The values of the termination symbols are a function of the encoder state after u_{4095} is clocked in; they are shown on the right in Figure 5. With termination, the CC encoder produces an output codeword, $\mathbf{c}_1$, consisting of 8196 symbols.

Puncturing and Interleaving. The next operations in Figure 4 are puncturing (the block denoted by the symbol $\mathcal{P}$) and interleaving (the block denoted by the symbol Π). The puncturing notation used below follows that given in [8]. The final coding rate of exactly $2/3$ is achieved by puncturing the rate-$1/2$ code as follows. We begin with the buffer $\mathbf{c}_1$, which consists of 8196 symbols and is shown in the lower part of Figure 6. The rate-$2/3$ puncturing pattern

$$\mathcal{P}_{2/3} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

is used to puncture (delete) every other symbol that came from the lower (parity) stream of the CC encoder (denoted by the 0 in the puncturing pattern), but leaves the other parity symbol and the two information bits intact. This pattern is repeated a total of 2042 times, which is nearly the entire length of $\mathbf{c}_1$. The rate-$3/4$ puncturing pattern

$$\mathcal{P}_{3/4} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

is used to puncture (delete) two out of every three parity symbols (denoted by the two 0's in the puncturing pattern), but leaves the other parity symbol and the three information bits intact. This pattern is applied a total of 4 times. As shown in Figure 6, this leaves only the termination symbols and their corresponding parity symbols. The two parity symbols associated with the termination symbols are punctured entirely. The net result of this puncturing scheme is that $2042 + 4 \cdot 2 + 2 = 2052$ symbols are deleted from $\mathbf{c}_1$. This leaves exactly $N = 6144$ symbols remaining, $K = 4096$ of which are information bits.

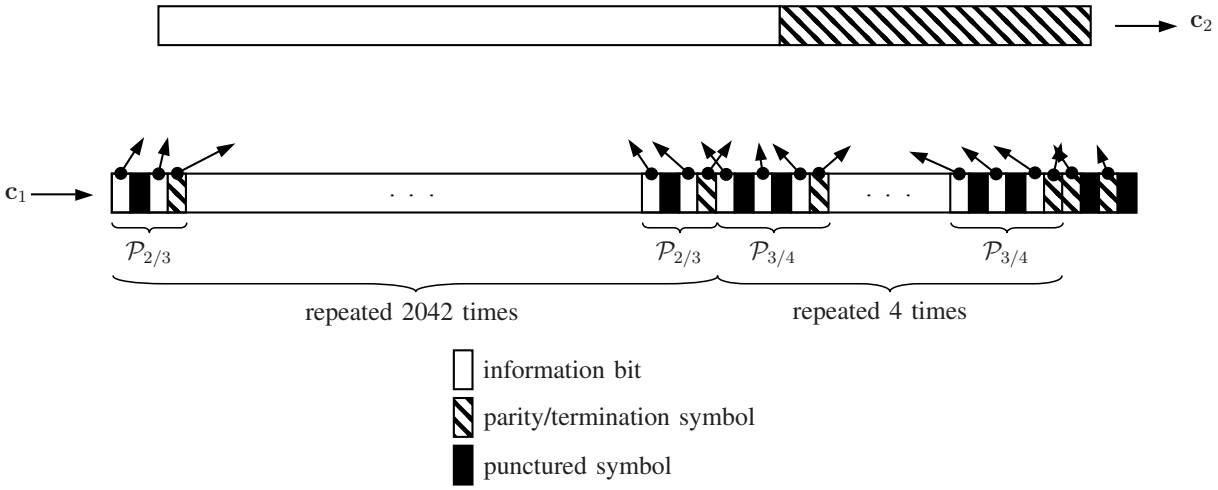


Figure 6: Block diagram of the puncturing and interleaving operations. The length-8196 input buffer, c_1, has 2052 symbols that are punctured (deleted). The 6144 non-punctured symbols in c_1 are randomly connected (with some constraints) to the 6144 locations in the output buffer, c_2. The most observable constraint is that the first 4096 positions in c_2 contain the $K = 4096$ information bits (in randomly permuted order) and the last 2048 positions in c_2 contain parity/termination symbols.

The interleaver output buffer, $\mathbf{c}_2$, is shown in the upper part of Figure 6 and contains exactly $N = 6144$ symbols. The connections between $\mathbf{c}_1$ and $\mathbf{c}_2$ are depicted by the many arrows leaving $\mathbf{c}_1$ and leading to $\mathbf{c}_2$. These connections are *random*[1] except for the following constraints:

- The symbols in $\mathbf{c}_1$ that are to be punctured have no connection to $\mathbf{c}_2$ (this obviously deletes the symbols as far as $\mathbf{c}_2$ is concerned).
- Any two adjacent symbols in $\mathbf{c}_1$ are separated by at least S locations in $\mathbf{c}_2$ (by *adjacent* we mean *adjacent non-punctured* symbols in $\mathbf{c}_1$). This constraint means that we are building an S-random interleaver [9].
- The information bits in $\mathbf{c}_1$ are located in the first $2/3$ of $\mathbf{c}_2$, and the parity/termination symbols are located in the last $1/3$ of $\mathbf{c}_2$. This is depicted by the shading of $\mathbf{c}_2$ Figure 6.

The last constraint is basically optional; however, it gives $\mathbf{c}_2$ the desired systematic format of Figure 2, which it then shares in common with the LDPC code word. As such, we refer to this interleaver as a *systematic S-random interleaver*. It is important to note that the last constraint can be worked into the interleaver design algorithm in [9] without interfering with the S-random constraint. In fact, we were able to find a $N = 6144$ interleaver with $S = 57$, which is slightly higher than the expected upper value of $\sqrt{6144/2} \approx 55.4$ [9]. Although more intricate in its *design*, the systematic S-random interleaver is no more complicated at *run-time* than an ordinary S-random interleaver.

Differential encoding. The last operation in Figure 4 is differential encoding. We use the differential encoding rule known as *double differential encoding* [10]. Internally, the encoder has a generic input of $\mathbf{u}$ and a generic output of $\mathbf{c}$. In our application, we drive the encoder with $\mathbf{u} = \mathbf{c}_2$, and the output is the final SCCC output of $\mathbf{c}$. The input/output relationship of this encoder is

$$c_i = u_i \oplus c_{i-2}, \qquad u_i, c_i \in \{0, 1\} \tag{1}$$

with $\oplus$ denoting the logical XOR operation. The differential encoder (DE) can be viewed as a $R = 1$ recursive convolutional code. We define the even-indexed inputs as $\{u_i\}_{i\text{-even}}$ and the odd-indexed inputs as $\{u_i\}_{i\text{-odd}}$. The double differential encoder can be thought of as two separate *single* differential encoders, where one encodes the even-indexed inputs and the other encodes the odd-indexed inputs.

[1]The *randomness* occurs only as the interleaver is being designed. There is no randomness when the interleaver and de-interleaver are being used; they are known to both the transmitter and receiver and remain fixed.

Unlike the CC encoder, where we assumed the encoder was initialized to the all-zeros state, we assume here that the encoder is initialized with $c_{-2} = 1$ and $c_{-1} = 0$, which is done for reasons of compatibility with the SOQPSK modulator that follows. Also, this encoder is not terminated to a known state, which results in lower-quality decoding estimates for the last few symbols; however, this uncertainty is widely distributed over the entire $N = 6144$ code word via the de-interleaver and thus has a negligible impact on the overall performance of the code.

Because the system in Figure 4 is the concatenation of two constituent codes, separated by an interleaver, it is a serially concatenated convolutional code in its own right. The CC encoder is the *inner code* and the DE is the *outer code* (or the DE plus the SOQPSK modulator). When the modulation itself is fully included in the decoding loop, we are able to achieve the best overall performance, as shown later. However, even when the modulation is ignored in the decoding loop, we are still able to achieve very good performance. The flexibility of including/not including the modulation in the decoding loop is a key contribution of this paper and broadens the number of demods that can be used with FEC systems.

Another key contribution of this paper is to have designed a SCCC code word that is formatted as closely as possible to the LDPC code word. Because test ranges are outfitted with a heterogenous mix of transmitters and receivers/decoders, it is possible to have a ground station that is not equipped with an FEC decoder. This is because FEC decoders are expensive, and less widely deployed than legacy equipment.

The common (and systematic) format of the two types of FEC code words makes them more friendly for an environment with this heterogeneous mix of equipment. In the event that a SCCC code word is received by a ground station without an FEC decoder, the information bits can be extracted via differential decoding and deinterleaving operations, both of which are trivial compared to a full-blown FEC decoding operation. For LDPC code words received by a terminal lacking a decoder, the information bits are simply retained as is, and the parity symbols are discarded. The coding gains afforded by the FEC encoding are forfeited in either case; however, interoperability with simple diversity receivers is very beneficial.

FEC DEMODULATOR/DECODER SYSTEMS

The FEC systems require the use of a *soft-output* demodulator. For demodulators that use a trellis, we have implemented the *soft-output Viterbi algorithm* (SOVA) [11]. For symbol-by-symbol (SxS) demodulators, the soft output is obtained simply by using the raw detection filter (DF) output. In addition to the demodulator SOVA, we also need a SOVA for the convolutional code (CC SOVA) and for the differential encoder (DE SOVA).

The two major types of demodulators (SxS and trellis) and two types of FEC codes (LDPC and SCCC) can be combined to form four distinct demodulator/decoder systems, the block diagrams of which are shown in Figures 7–10. Before discussing how each system is different from the others, we first concentrate on the attributes they all have in common. In fact, this is one of the main contributions of this paper: to design FEC systems that are as similar as possible. In particular,

- Each system has a "stand-alone" demodulator that is completely decoupled from the FEC decoder. The demodulator operates on the received signal, $r(t)$, performs all synchronization tasks, and returns an initial soft estimate, $\hat{\mathbf{c}}_0$, of the codeword.

- Each system has a "stand-alone" FEC decoder that is completely decoupled from the soft-output demodulator. The FEC decoder operates on the demodulator output and returns the estimated information word, $\hat{\mathbf{u}}$.

- Because the codewords are systematically encoded, and because FEC decoders are relatively costly, in the event the receiver is not equipped with a FEC decoder, $\hat{\mathbf{u}}$ can be recovered from a hard-limited version of $\hat{\mathbf{c}}_0$ with relative ease (but with no FEC performance gain, of course). For the LDPC systems in Figures 7 and 8, $\hat{\mathbf{u}}$ occupies the first 4096 positions of $\hat{\mathbf{c}}_0$. For the SCCC system in Figure 9, $\hat{\mathbf{u}}$ is a 4096-bit subset of a de-interleaved version of $\hat{\mathbf{c}}_0$. For the SCCC system in Figure 10, $\hat{\mathbf{u}}$ is a 4096-bit subset of a differentially decoded and de-interleaved version of $\hat{\mathbf{c}}_0$.

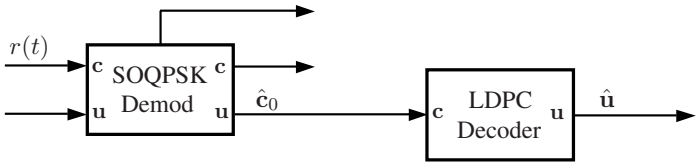


Figure 7: Block diagram of the LDPC system with a trellis demodulator. There is no differential encoding in this system.

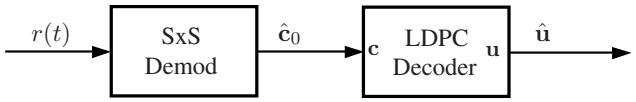


Figure 8: Block diagram of the LDPC system with a SxS demodulator. There is no differential encoding in this system.

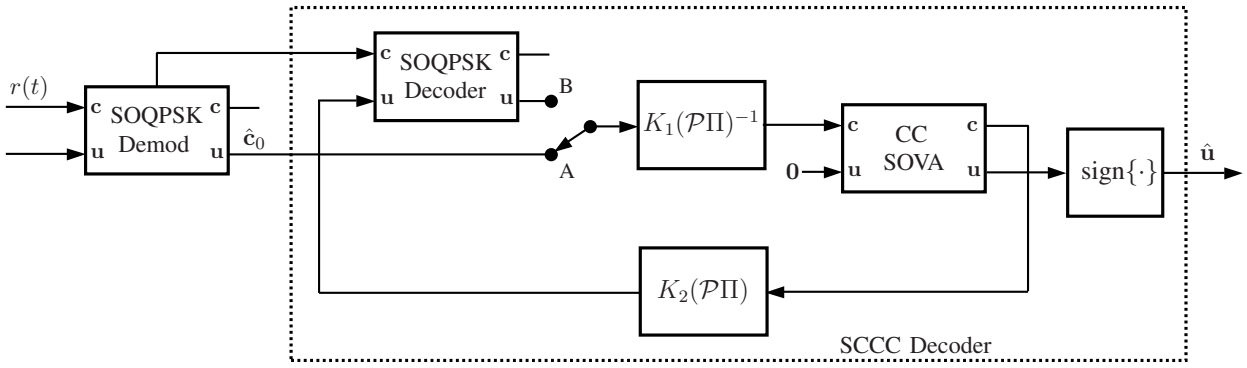


Figure 9: Block diagram of the SCCC system with a trellis demodulator. There is differential encoding in this system.

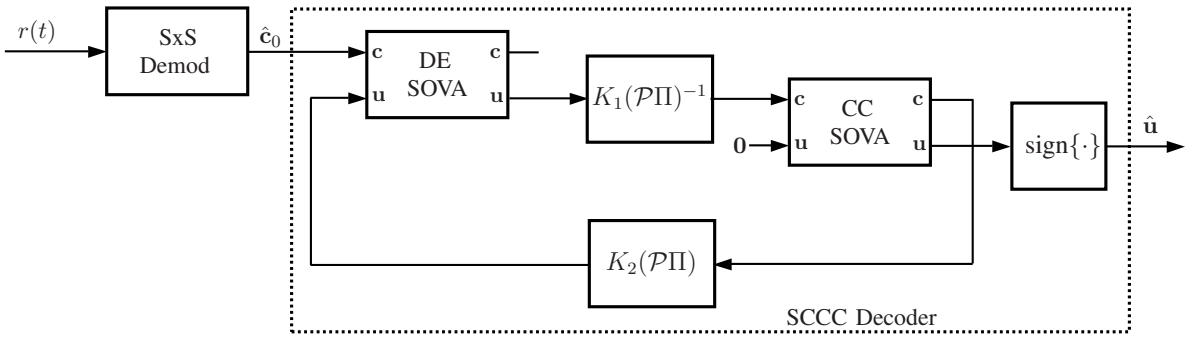


Figure 10: Block diagram of the SCCC system with a SxS demodulator. There is differential encoding in this system.

The two LDPC systems in Figures 7 and 8 are pretty straightforward in the way they are connected, so no additional explanation is needed. On the other hand, the "SCCC decoders" (i.e. the enclosed sub-diagrams in Figures 9 and 10) need some additional explanation. The specifics of the SCCC decoder depend on which demodulator is being used.

When the trellis demodulator is being used (Figure 9), the "SOQPSK Demod" does the first part of the SCCC decoding process. The switch in Figure 9 is initially in position "A." The demodulator output is fed to a block that scales it by a factor of $K_1 = 0.75$, de-interleaves it, and de-punctures it; these operations are signified by $K_1(\mathcal{P}\Pi)^{-1}$. The CC SOVA does its processing next. Once it is finished, its upper output is scaled by $K_1 = 0.75$, re-interleaved, and re-punctured $[K_2(\mathcal{P}\Pi)]$. The SCCC decoder is then ready to commence its second iteration. The switch is now placed in position "B." The "SOQPSK Decoder" is now used for the first time and the remainder of the second iteration proceeds as with the first. The "SOQPSK Decoder" *reuses* the matched filter (MF) outputs from the "SOQPSK Demod," and thus it does not have to redo the synchronization tasks of the demod. When N_{it} iterations are finished, the lower output of the CC SOVA is hard-limited to produce $\hat{\mathbf{u}}$. Although the SCCC decoding process is shown as a loop, it can also be implemented in a pipelined architecture for to yield a much higher decoder throughput.

When the SxS demodulator is being used (Figure 10), the demodulator does no part of the decoding process. Here, the first SCCC decoding iteration begins with the DE SOVA, the lower input to which is set to $\mathbf{0}$ initially. The remainder of the SCCC decoding steps are as explained above. Because the actual inner code is best modeled by the DE plus the SOQPSK modulator, the system in Figure 10 incurs a small performance penalty, as shown below.

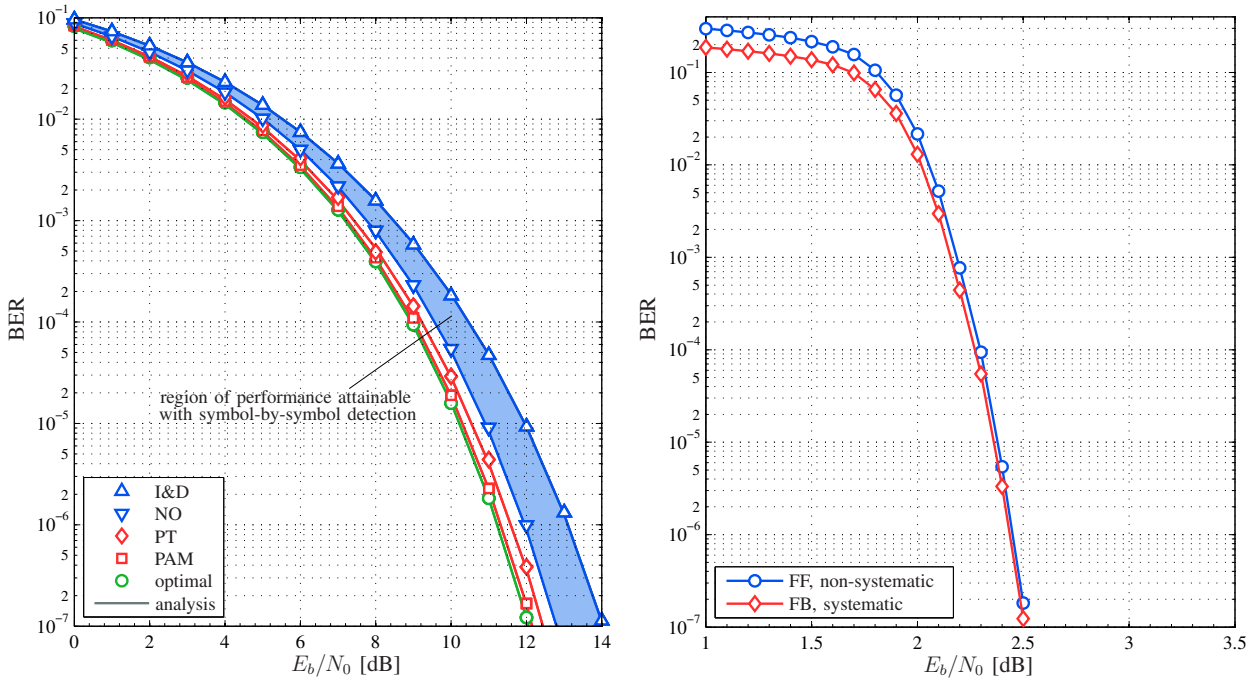
BIT ERROR RATE PERFORMANCE OF THE FEC SYSTEMS

In this section we provide numerical results on the bit error rate (BER) performance of the systems described above. In most instances, we give performance data for four different demodulators: trellis demod with PAM filters [12], trellis demod with PT filters [12], SxS demod with a numerically-optimized (NO) filter [13], and SxS demod with an integrate-and-dump (I&D) filter. In some instances, however, we give data only for the trellis demod with PT filters. Also, reference data is sometimes available for the optimal demodulator—maximum likelihood sequence detection (MLSD) with a 512-state trellis [12].

Uncoded Systems

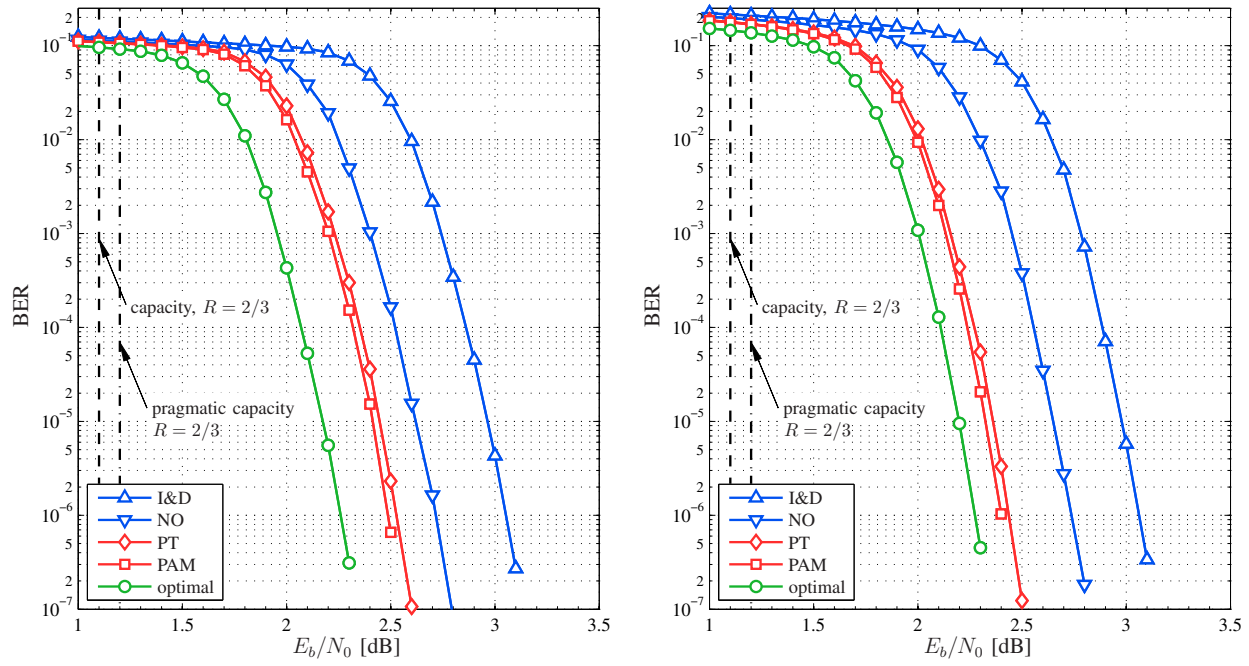
Figure 11 (a) shows the performance of uncoded systems (without differential encoding), i.e. the stand-alone performance of the demodulators. In each case, we show a solid curve for the analytical probability of bit error (P_b), and discrete BER simulation points plotted on top of these curves. For the optimal detector, the analytical P_b curve is given by $P_b = 1/2Q(\sqrt{1.60E_b/N_0}) + 1/2Q(\sqrt{2.59E_b/N_0})$. With SOQPSK, there are basically two "error modes." The first is where two competing data sequences deviate by 90° at some symbol interval, then remain apart by 90° during the following symbol interval, and then merge back together during the third symbol interval. This error mode has a normalized squared Euclidean distance (NSED) of 1.60. The second error mode is where two competing data sequences deviate by 90° at some symbol interval, then each go 90° in the opposite direction during the following symbol interval (ending up 90° apart again), and then merge back together during the third symbol interval. This error mode has a NSED of 2.59. Both error modes have an equal number of Tx/Rx sequences where they can occur. At low E_b/N_0, both modes occur frequently; however, as E_b/N_0 increases, the second error mode quickly becomes rare.

The 4-state trellis demods suffer very little performance loss relative to the optimal 512-state demod, as was shown in [12]. As expected, the trellis-based demods outperform the SxS demods; however, the SxS NO has very attractive performance given its simplicity.



(a) (b)

Figure 11: (a) BER results for *uncoded* SOQPSK-TG, without differential encoding. (b) BER results comparing the proposed FB, systematic SCCC format, against the conventional FF, non-systematic SCCC format.



(a) (b)

Figure 12: BER results for a variety of SOQPSK-TG demodulators paired with (a) LDPC and (b) SCCC.

Proposed SCCC Design vs. Original SCCC Design

An important question is whether or not the SCCC enhancements described earlier (Figure 6) are worth the design effort, i.e. is performance better, the same, or worse than the conventional SCCC system in [12]? For the sake of clarity, the conventional system in [12] uses a non-systematic, feed-forward (FF) CC encoder, no trellis termination, and a non-systematic interleaver.

This question is answered in Figure 11 (b). The conclusion is that at low E_b/N_0 (high BER), the proposed system is better, and for asymptotically large E_b/N_0 (low BER) there is no difference. Although results are shown for only one demod example, there is no reason to expect different results for the other demods. Given the system design benefits (compatibility with demods without FEC decoders, greater compatibility with the LDPC format), no complexity disadvantages, and no performance disadvantages, the proposed SCCC design is recommended for adoption.

LDPC and SCCC with Various Demodulators

Figure 12 (a) shows the performance of LDPC-encoded SOQPSK-TG with the four demod examples. The LDPC decoder performs a maximum of $N_{\text{it}} = 200$ iterations. A reference curve is also given for the optimal demod with the optimal LDPC decoding algorithm [6, Algorithm 15.2]. There is a 0.2 dB loss that is attributable to the scaled-min decoding algorithm [14] that we used. In addition to this, it is notable that the BER curves are more tightly bunched in the FEC case; that is to say, the losses in Figure 12 (a) are smaller for the progressively suboptimal demods than they are in Figure 11 (a). This result makes an even stronger case for the use of the simple-yet-robust SxS detectors: the 0.2 dB loss of the NO SxS detector is a fair trade for the simplified implementation.

Figure 12 (b) shows the performance of SCCC-encoded SOQPSK-TG with the four demod examples. The SCCC decoder performs $N_{\text{it}} = 16$ iterations. A reference curve is also given for the optimal demod with the optimal soft-input soft-ouput (SISO) algorithm [15] used in place of the SOVA. Comparing the optimal cases, LDPC has a slight advantage over SCCC. However, for the suboptimal-but-practical cases, the SCCC systems have a slight advantage over LDPC for asymptotically large E_b/N_0 (low BER).

As with LDPC, the SCCC curves are more tightly bunched that in the uncoded case. However, the SxS detectors have a slightly larger loss with SCCC than with LDPC. This is attributable to the fact that the *symbol*-energy-to-noise ratio, E_s/N_0, in which the SxS demod in Figure 10 operates, is low enough that both SOQPSK error modes occur frequently. The DE SOVA that follows in Figure 10 does not have a large enough trellis to resolve these two error modes from each other as the decoding iterations progress. On the other hand, because the "SOQPSK decoder" in Figure 9 is fed with MF outputs that preserve the $\pm 90^\circ$ transitions of the SOQPSK waveform, some additional gains are possible as the decoding iterations progress. Once again, though, we point out that the SxS detectors have attractive performance given their simplicity.

Figure 12 also shows the channel capacity of SOQPSK-TG with a coding rate of $R = 2/3$. Because LDPC does not use the modulation itself in the decoding process, its limit is given by the "pragmatic capacity," as explained in [16]. SCCC, on the other hand, does include the modulation as part of the code, and thus its limit is given by the true channel capacity. The codes are within around 1 dB of capacity.

CONCLUSION

We have given a description of two types of FEC—LDPC and SCCC—and two major types of SOQPSK demodulators—trellis and symbol-by-symbol—and have shown how each is properly implemented in FEC demodulator/decoder systems. Based on the numerical results we have presented, we conclude that the proposed SCCC format is the one that should be adopted. We have also shown that LDPC and SCCC have essentially the same BER performance. We have also shown that symbol-by-symbol demodulators represent a very attractive option due to their strong performance in the coded systems and their overall simplicity and proven robustness in the field.

ACKNOWLEDGEMENT

This project is funded by the Test Resource Management Center (TRMC) Test and Evaluation/Science & Technology (T&E/S&T) Program through the U.S. Army Program Executive Office for Simulation, Training and Instrumentation (PEO STRI) under Contract No. W900KK-11-C-0031.

REFERENCES

[1] integrated Network Enhanced Telemetry (iNET) Program, "Communication Links Standards Working Group (CLSWG) Face-to-Face Meeting," Pasadena, CA, Mar. 17–18 2008.

[2] K. S. Andrews, D. Divsalar, S. Dolinar, J. Hamkins, C. R. Jones, and F. Pollara, "The development of turbo and LDPC codes for deep-space applications," *Proc. IEEE*, vol. 95, pp. 2142–2156, Nov. 2007.

[3] Consultive Committee for Space Data Systems (CCSDS), "Low density parity check codes for use in near-Earth and deep space applications (131.1-O-2 Orange Book)," Sep. 2007.

[4] D. Kumaraswamy, "Simplified detection techniques for serially concatenated coded continuous phase modulations," Master's thesis, Dept. Elect. Eng. Comp. Sci., Univ. Kansas, Lawrence, KS, Aug. 2007.

[5] K. Damodaran, "Serially concatenated coded continuous phase modulation for aeronautical telemetry," Master's thesis, Dept. Elect. Eng. Comp. Sci., Univ. Kansas, Lawrence, KS, Dec. 2008.

[6] T. K. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. New York: Wiley-Interscience, 2005.

[7] A. V. Oppenheim and A. S. Willsky, *Signals and Systems*. New York: Prentice Hall, 1997.

[8] Y. Yasuda, K. Kashiki, and Y. Hirata, "High-rate punctured convolutional codes for soft decision Viterbi decoding," *IEEE Trans. Commun.*, vol. 32, pp. 315–319, Mar. 1984.

[9] D. Divsalar and F. Pollara, (1995, May), "Multiple turbo codes for deep-space communications," *Telecommunications and Data Acquisition Progress Report*, vol. [Online]. Available: http://tmo.jpl.nasa.gov/tmo/progress_report/42-121/121T.pdf.

[10] M. K. Simon, "Multiple-bit differential detection of offset QPSK," *IEEE Trans. Commun.*, vol. 51, pp. 1004–1011, Jun. 2003.

[11] M. P. C. Fossorier, F. Burkert, S. Lin, and J. Hagenauer, "On the equivalence between SOVA and max-log-MAP decodings," *IEEE Commun. Lett.*, vol. 2, pp. 137–139, May 1998.

[12] E. Perrins and M. Rice, "Reduced-complexity approach to iterative detection of SOQPSK," *IEEE Trans. Commun.*, vol. 55, pp. 1354–1362, Jul. 2007.

[13] M. Geoghegan, "Optimal linear detection of SOQPSK," in *Proc. Int. Telemetering Conf.*, Oct. 2002.

[14] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of ldpc codes," *IEEE Trans. Commun.*, vol. 53, pp. 1288–1299, Aug. 2005.

[15] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "A soft-input soft-output APP module for iterative decoding of concatenated codes," *IEEE Commun. Lett.*, vol. 1, pp. 22–24, Jan. 1997.

[16] C. Şahin and E. Perrins, "The capacity of SOQPSK-TG," in *Proc. IEEE Military Commun. Conf.*, (Baltimore, MD), pp. 555–560, Nov. 2011.