The Invention of Turbo Codes

Claude Berrou

This is a companion piece to my article "The Ten-Year-Old Turbo Codes are Entering into Service" which was published in *IEEE Communications Magazine* (vol, 41, no. 8, August 2003, DOI: 10.1109/MCOM.2003.1222726) and republished on the next page.

The institution where I spent my entire career was called, at its creation in 1977, École Nationale Supérieure des Télécommunications de Bretagne. Today the name is IMT Atlantique. I was recruited as a lecturer as soon as this institution was founded and for about 10 years, I devoted myself entirely to teaching semiconductor physics, digital and analog integrated circuits, and microwaves. It was exciting, and research did not seem necessary to me, even if of course constant technology survey was indispensable.

In 1988, a colleague I did not know very well asked me for a meeting. He was Alain Glavieux, a recognized expert in digital communications who was working at that time on the improvement of an underwater acoustic video transmission system. This system had already been used to observe the wreck of the Titanic, but its quality needed to be improved.

Alain explained to me in detail what he was proposing to study. The surface receiver contained an equalizer followed by an error correcting decoder, both functions based on the Viterbi algorithm. But the equalizer could only provide the decoder with binary decisions. I was therefore asked to imagine an equalization circuit, as simple as possible, able to provide the decoder with weighted decisions. These probabilistic data would increase the correction power of the decoder, and a gain of about 2 dB could be expected in the link budget.

At that time, the subject was topical, and two publications, one by Gérard Battail [1] and the other by Joachim Hagenauer and Peter Hoeher [2], were precious starting points for me. I knew the Viterbi algorithm because I had supervised students in the design of a gate-array integrated circuit on this subject. However, my expertise was still limited.

So here I was, spending days in convolutional code lattices, in the company of probability logarithms and under the double injunction of good performance and simplicity. This work was certainly the most intense of all that I undertook. A convincing result was finally obtained, with the help of my colleague Patrick Adde, and was presented at ICC 1993 [3].

This was also the period when some experts were questioning the validity of the limits established by Claude Shannon or, at least, the reasons why they were inaccessible. The best performing code that had been imagined until then was a concatenated code: a Reed-Solomon encoder followed by a convolutional encoder. About 3 dB kept it away from the theoretical limit.

Others [4] before me had observed that the corresponding decoder was suffering from a strong asymmetry. The outer decoder (according to the Berlekamp-Massey algorithm, for example) can benefit not only from the redundancy produced by the Reed-Solomon code but also from the work of the inner decoder, and thus indirectly from the redundancy produced by the convolutional code. The reverse is not true: the inner decoder does not benefit from the outer decoder's power of correction.

It seemed to me that this asymmetry could be easily corrected if the Reed-Solomon code was replaced by a second convolutional code. The weighted output Viterbi decoder was now available to implement near-optimal feedback.

A few days were enough to validate this idea, and the rest went very quickly: introduction of extrinsic information (the concept had already been introduced in [5], a publication I did not know), invention of recursive systematic convolutional (RSC) codes, and replacement of serial concatenation by parallel concatenation. On this last point, the motivation was linked to my intention to concretize this new coding/decoding scheme by an integrated circuit that would be designed in part by my students. Parallel concatenation simplifies considerably the specifications in that only one clock is needed to drive all the elements of the circuit, as opposed to two decoders that work with distinct rates.

Meanwhile, Alain Glavieux and I had become very friendly. One of his doctoral students, Punya Thitimajshima, a Thai national, started to study recursive systematic convolutional codes and made it his main thesis topic. He was also interested in an algorithm that I did not know: the maximum a posteriori (MAP) algorithm [6]. Alain and his Ph.D. student pointed out to me that what separated the performance of my new coding/decoding scheme from Shannon's limits could be reduced by replacing the output-weighted Viterbi algorithm with this MAP algorithm. I therefore got interested in it, without being really convinced by the possibility of making a version that could be implemented on silicon. Thus, the patent filed in 1991 on the parallel concatenation of RSC codes and its iterative decoding mentioned the Viterbi decoding with weighted output, while the publication, two years later and under the name of turbo code, was based on the MAP algorithm. I salute the memory of Alain and Punya, who were taken from us too soon.

Then came the time of invited conferences and awards. This gave me the opportunity to meet a lot of colleagues, most of whom I was discovering as I was taking my first steps in this vast international coding and communications community.

Scientific research is not practiced in a world without relief, fortunately. The diversity of approaches and actors is to be respected and promoted, whether they are experimenters like Marconi or Tesla, or theorists like Poincaré or Einstein. Let us keep in mind what the latter of these great scientists observed and of which I became convinced: "We can't solve problems by using the same kind of thinking we used when we created them." In this state of mind, interdisciplinarity is a valuable path, albeit a somewhat perilous one, because it is not always fully appreciated by expert committees.

References

[1] G. Battail, "Pondération Des Symboles Décodés par L'Algorithme de Viterbi," *Annales des Telecommunications* (in French), vol. 42, Jan. 1987, pp. 31–38.

[2] J. Hagenauer and P. Hoeher, "A Viterbi Algorithm with Soft-Decision Outputs and Its Applications," *Proc. IEEE GLOBECOM '89*, Dallas, TX, Nov. 1989, pp. 47.11–47.17.

[3] C. Berrou *et al.*, "A Low Complexity Soft-Output Viterbi Decoder Architecture," *Proc. IEEE ICC '93*, 1993, vol. 2, pp. 737–40.

[4] J. Hagenauer, E. Offer, and L. Papke, "Improving the Standard Coding System for Deep Space Missions," *Proc. IEEE ICC '93*, Geneva, Switzerland, vol. 2, May 1993, pp. 1092–97.

[5] R. Gallager, "Low-Density Parity-Check Codes," *IRE Trans. Info. Theory*, vol. 8, no. 1, Jan. 1962, pp. 21–28.

[6] L. Bahl *et al.*, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate (Corresp.)," *IEEE Trans. Info. Theory*, vol. 20, no. 2, Mar. 1974, pp. 284–87.

CAPACITY APPROACHING CODES, ITERATIVE DECODING ALGORITHMS, AND THEIR APPLICATIONS

The Ten-Year-Old Turbo Codes are Entering into Service

Claude Berrou, ENST Bretagne

ABSTRACT

In the matter of channel coding and spectral efficiency, up to the invention of turbo codes, 3 dB or more stood between what the theory promised and what real systems were able to offer. This gap has now been removed, allowing communication systems to be designed with quasi-optimum performance. Ten years after the first publication on this new technique, turbo codes have commenced their practical service.

INTRODUCTION

What is clear today is that Claude Shannon did not make the slightest mistake when he calculated the potential of channel coding and his famous capacity limits. We are now able to attain these limits, by some hundredths of a decibel [1, 2], thanks to turbo codes or turbo-like codes, but in no case, apparently, to go beyond them. A real barrier! And how laborious it was to come close to this asymptote! Hamming, Elias, Reed and Solomon, Bose, Chaudhuri and Hocquenghem, Gallager, Berlekamp, Forney, Viterbi, and so many others made important contributions to the 50-year-old edifice. But there seemed to be a latent prejudice in the field of information theory: because the foundation of digital communications relied on potent mathematical considerations [3], error correcting codes were believed to belong solely to the world of mathematics. If it is true that good codes (BCH, Reed-Solomon, etc.) were defined thanks to algebraic tools, physics also had its say in this story, in particular regarding decoding techniques. Turbo decoding was devised in this spirit, with the permanent intuition that the feedback concept, so precious in electronics, for instance, could also contribute to the decoding of compound (concatenated) codes, and this was indeed the doorway to iterative decoding. The issue of stability, which is crucial in feedback systems and thus in turbo decoders, was easily solved by introducing the notion of extrinsic information, which prevents the cascade decoder from being a positive feedback amplifier. Another concern was the search for symmetry, a basic rule in many physical structures, and this concern led to the so-called parallel concatenation concept, which offers a perfect balance between the component codes, unlike classical serial concatenation.

Turbo codes were presented to the scientific community just 10 years ago [4]. Their invention was the result of a pragmatic construction conducted by C. Berrou and A. Glavieux [5], based on the intuitions of G. Battail, J. Hagenauer, and P. Hoeher, who, in the late ’80s, highlighted the interest of probabilistic processing in digital communication receivers [6,7]. Previously, other researchers, mainly P. Elias [8], R. Gallager [9], and M. Tanner [10], had already imagined coding and decoding techniques whose general principles are closely related to those of turbo codes. Since 1993, turbo codes have been widely studied and adopted in several communication systems, and the inherent concepts of the “turbo” principle have been applied to topics other than error correction coding, such as single-user and multi-user detection.

BRIEFLY, HOW DOES IT WORK?

First, let us go back 30 years in the history of coding. In a well-known paper [11], D. Forney made an in-depth presentation of convolutional codes, which can take the two forms described in Fig. 1, with the example of $\nu = 3$ memory units: nonrecursive nonsystematic (a), and recursive systematic (b). For reasons that are not so obvious today, with the passing of time, D. Forney advocated the use of the first structure, which has indeed been widely used with success in many digital transmission systems. Turbo codes use the other structure, which offers several advantages in comparison with the former. The first is conceptual: a recursive systematic convolutional (RSC) code is based on a pseudo-random scrambler, and actually, random codes were used by Shannon to calculate the theoretical potential of channel coding. The second advantage is decisive for high coding rates and/or high levels of noise: it just works better. The final advantage, related to the previous one and also fundamental for turbo coding, concerns what are called *return to zero sequences*. This is developed in the next paragraph.

RTZ Sequences and Circular Encoding

Suppose that both registers of nonrecursive and recursive encoders, each having ν memory units, are initialized in state 0, and that any random sequence, followed by some (at least ν) additional 0s, feeds the two encoders. After the full sequence has passed, the nonrecursive encoder systematically returns to state 0, whereas the recursive encoder does so with probability $1/2^\nu$. This is because the latter is based on a pseudorandom scrambler. When the register goes back to state 0 after the encoding of a given sequence followed by some 0s, this sequence is called a return to zero (RTZ) sequence. So all random sequences continued by at least ν 0s are RTZ for a nonrecursive encoder, and only a fraction ($1/2^\nu$) of them are RTZ for a recursive encoder.

Now, let us adapt the convolutional code in order to transform it into a block code. The best way to do this is to allow any state as initial state and to encode the sequence, containing k information bits so that the final state of the encoder register will be equal to the initial state. The trellis of the code (the temporal representation of the possible states of the encoder, from time $i = 1$ to $i = k$) can then be regarded as a circle, and this technique is called *circular* (or *tail-biting*) termination. In the sequel we call the circular version of RSC codes *circular RSC* (CRSC). Thus, without having to pay for any additional information, and therefore without impairing spectral efficiency, the convolutional code has become a real block code, in which, for each time i, the past is also the future, and vice versa. This means that a non-RTZ sequence produces effects on the whole set of redundant symbols stemming from the CRSC encoder, around the whole circle; and thanks to this very informative redundancy, the decoder has very little probability of failing to recover this non-RTZ sequence. This property explains the superiority, in most situations, of CRSC codes over classical nonrecursive convolutional codes, for which all sequences are RTZ. It is also the key for constructing compound CRSC codes, because a circular trellis does not display any side effect that could be detrimental when aiming at low error rates.

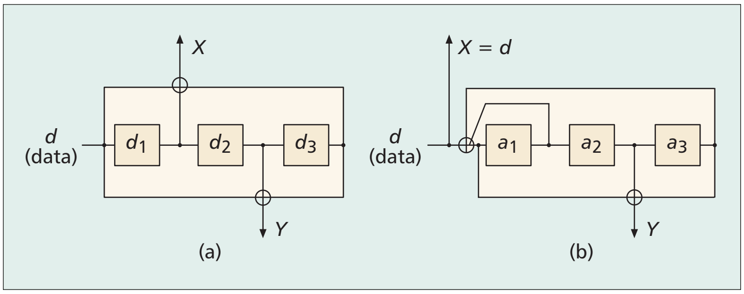

Figure 1. *a) Classical nonrecursive nonsystematic convolutional code with $\nu = 3$ memory units (8-state); b) the equivalent recursive systematic version of the previous one, which is the building block for turbo encoding.*

Parallel Concatenated CRSC Codes and Turbo Codes

We have seen that the probability that any given sequence is an RTZ sequence for a CRSC encoder is $1/2^\nu$. Now, if we encode this sequence N times (Fig. 2a), each time in a different order,

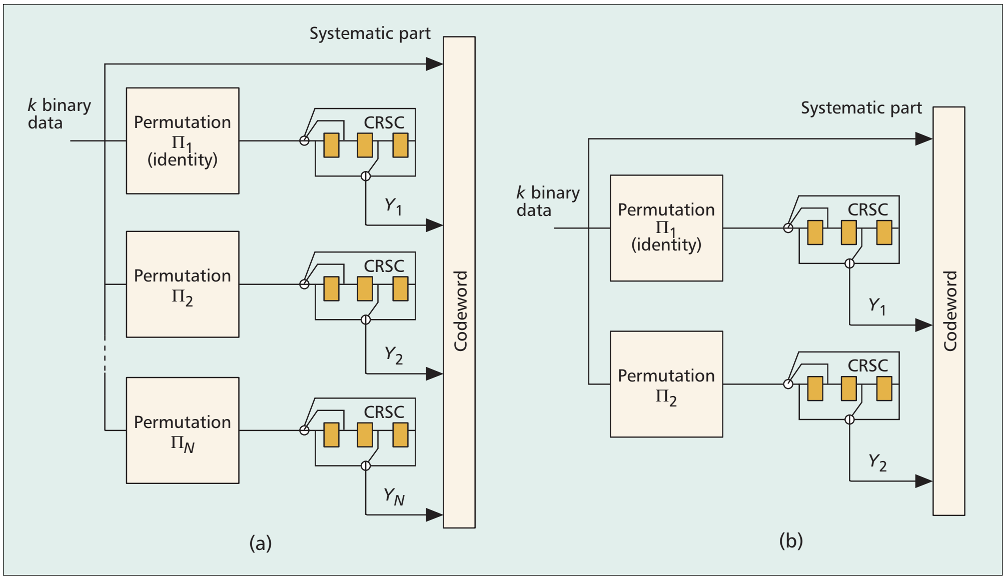

Figure 2. *a) In this multiple parallel concatenation of CRSC codes, the block containing* k *information bits is encoded* N *times. The probability that the sequence remains of the RTZ type after the* N *permutations, drawn at random (except the first one), is very low. The properties of this multiconcatenated code are very close to those of random codes. b) The number of encodings can be limited to two, provided that permutation Π_2 is judiciously devised. This is a classical turbo code.*

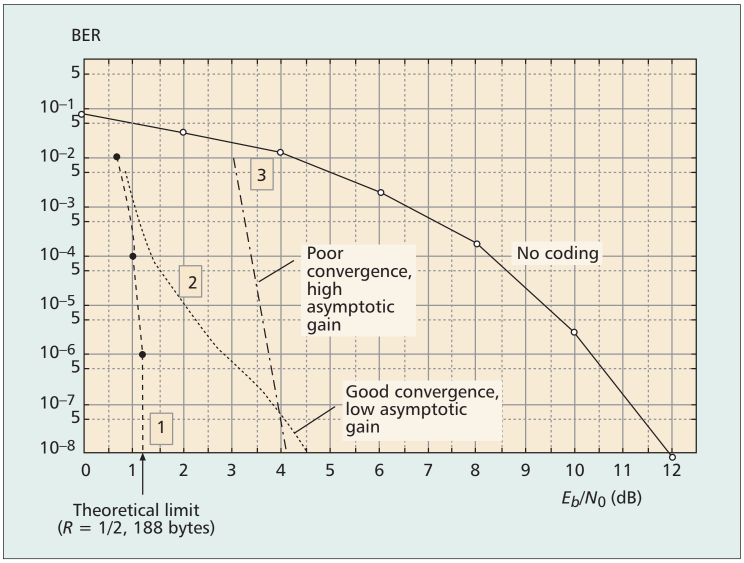

Figure 3. *Possible behaviors for a coding/decoding scheme on a Gaussian channel.*

drawn at random (the first order may be the natural order), the probability that the sequence remains RTZ for all encoders is lowered to $1/2^{N\nu}$. For instance, with $\nu = 3$ and $N = 7$, this probability is less than 10^{-6}. In other cases, at least one encoder will deliver sufficient redundant information for the decoder to retrieve the proper sequence. This technique is known as a multiple parallel concatenation of CRSC codes. Fortunately, it is possible to obtain quasi-optimum performance with only two encodings (Fig. 2b), and this is a classical turbo code. For bit error rates (BER) higher than around 10^{-5}, the permutation may still be drawn at random, but for lower BERs a particular effort has to be made in its design. The way the permutation is devised fixes the minimum Hamming distance $d_{\min}$ of the turbo code, and therefore the achievable asymptotic signal-to-noise ratio gain G_a, according to the well-known approximation, for a coding rate equal to R, under soft decision decoding:

$$G_a(\text{dB}) \approx 10\log_{10}(Rd_{\min}).$$

Turbo Decoding

Turbo decoding relies on the following fundamental criterion, which is applicable to all message passing or belief propagation [12] algorithms:

When several probabilistic machines work together on the estimation of a common set of symbols, all the machines have to give the same decision, with the same probability, about each symbol, as a single (global) decoder would.

To achieve this, turbo decoding, or any turbo process in general, relies on the exchange of probabilistic messages between all the processors dealing with the same data. For instance, the decoding of the classical turbo code illustrated in Fig. 2b involves using two processors, namely two soft-in/soft-out (SISO) decoders, that one could also call probabilistic decoders. Each decoder processes its own data, and passes the so-called extrinsic information to the other decoder. Usually, but not necessarily, computations are done in the logarithmic domain. Denoting $\Pr\{d = 1\}$ the probability that, at a given level of an evaluation processus, a particular binary datum is equal to logical "1," we write

$$L(d) = \ln\left(\frac{\Pr\{d=1\}}{1-\Pr\{d=1\}}\middle|\begin{matrix}\text{received}\\ \text{sequence}\end{matrix}\right).L$$

L is called the logarithm of the likelihood ratio. Then extrinsic information related to d is very simple to express as

$$L_{\text{extrinsic}}(d) = L_{\text{output}}(d) - L_{\text{input}}(d),$$

that is, what is passed by one decoder to the other is the result of its work about the estimation of d, but not taking its own input into account. The reason for this is that the input related to d is a piece of information shared by both decoders, and does not have to be a matter of additional information transfer.

When implemented in a digital circuit, turbo decoding is an iterative process, one iteration corresponding to one passing through each of all processors concerned.

The Assets of Turbo Codes

In light of the most recent results, how can we compare turbo codes to the ideal? In the following, we propose some answers regarding the main features that can be considered as characterizing a coding/decoding (codec) scheme: performance, decoding complexity, latency, and versatility.

About Performance

The performance of a codec depends on the type of perturbation (Gaussian, fading, impulsive, etc.), the coding rate, and the length of the block containing the data. It is generally accepted that the comparison that can be made between different codecs on an additive white Gaussian noise (AWGN) channel gives a hierarchy that is respected for other kinds of channels. So we will limit our discussion to this former case, which is also important in practice.

Figure 3 represents, as well as performance without coding, three possible behaviors for an error correcting coding scheme on an AWGN channel with binary phase shift keying (BPSK) or quaternary PSK (QPSK) modulation and coding rate 1/2. To be concrete, the information block is assumed to be around 188 bytes (MPEG application). The error probability P_e of the *no coding* performance is given by the complementary error function, as a function of E_b/N_0, where E_b is the energy per information bit and N_0 is the one-sided noise power spectrum density.

Case 1 corresponds to the ideal Shannon system, with the theoretical limit around 1.2 dB for a BER of 10^{-8}. Case 2 has good convergence and low $d_{\min}$. This is, for instance, what is obtained with turbo coding when the permutation function is not properly designed. Good convergence means that the BER decreases noticeably, close to the theoretical limit, and low $d_{\min}$ brings a severe change in the slope of the

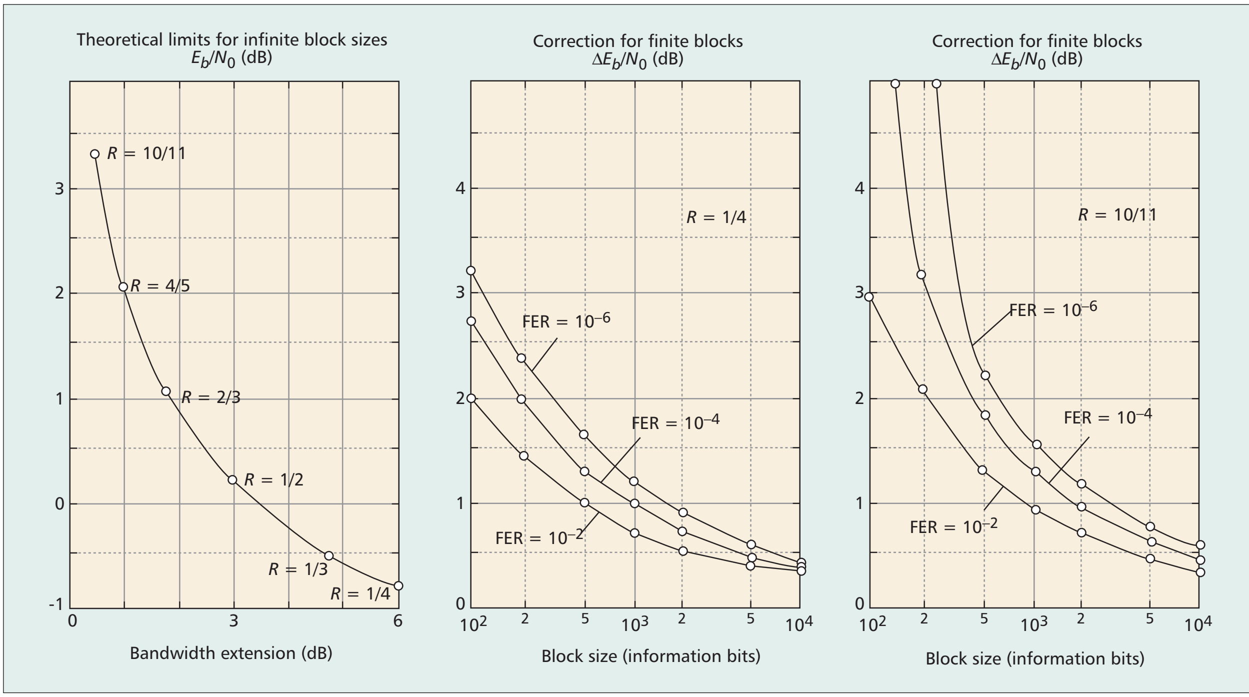

Figure 4. *The left curve represents the theoretical limit, in* E_b/N_0, *as a function of the coding rate or the equivalent bandwidth extension for infinite information blocks, a Gaussian channel, and binary modulations (BPSK, QPSK, etc.). The right curves provide the correction to add to the previous values to take the block size and target frame error rate (FER) into account. This is given for the two extreme coding rates of the left curve:* R = *1/4 and* R = *10/11.*

error performance curve due to a too small asymptotic gain. This gain is around 7.5 dB in the example given, and is reached at medium BER ($\approx 10^{-7}$). Below that, the curve remains parallel to the no coding one.

Case 3 has poor convergence and large $d_{\min}$. This is representative of a decoding procedure that does not take advantage of all the information available at the receiver side. A typical example is the classical concatenation of a Reed-Solomon code and a simple convolutional code. Whereas the minimum distance may be very large, the decoder is clearly suboptimal because the convolutional inner decoder does not take advantage of the RS redundant symbols.

The search for the perfect coding/decoding scheme has always faced the convergence vs. $d_{\min}$ dilemma. Usually, improving either aspect in some more or less relevant way weakens the other.

Turbo codes have constituted a real breakthrough with regard to the convergence problem. The ability of turbo codes to achieve near-optimum performance has revived the interest of system designers in the theoretical limits, which had flagged because of the difficulty of finding optimal coding. Some of these limits are represented in Fig. 4 for infinite and finite block sizes. One can particularly observe the non-negligible effect of the block size on these limits, especially for blocks having less than 10^3 information bits, and also for high coding rates. Furthermore, it is worth noticing that these limits are also dependent on the target frame error rate (FER) when the block has finite length. These limitations, which were not topical 10 years ago, are now of prime importance in the design of channel coding.

In conclusion, we grant that turbo coding and decoding are quasi-optimal in terms of convergence. But this is not the case regarding $d_{\min}$, which does not increase linearly with the block size as it would with virtual Shannon codes. The minimum distances of turbo codes are far from comparable to those of random codes. This is why it is not straightforward, but not impossible, to design turbo codes whose behavior is closer to situation 1 rather than situation 2 in Fig. 3.

About Decoding Complexity

Turbo decoding bypasses the exponential complexity of maximum likelihood decoding by means of an iterative procedure. To give an idea of the complexity requirement, let us make two concrete comparisons:

• Code A is a single RSC code with $\nu \approx 30$, providing a minimum distance around 30 too, for a coding rate of 1/2. Decoding this code by the now well-known *maximum a posteriori* (MAP) algorithm [13] would offer an asymptotic gain of about 12 dB. The number of states to consider in the trellis for each information bit is 2.10^9. The factor 2 is added because the MAP algorithm requires backward and forward processes. Code B is a turbo code combining two 16-state RSC component codes. The convergence and minimum distance, and therefore the asymptotic gain, would be comparable to those of code A. The equivalent number of states to consider in total for each information bit would

The latency issue is a weak point for turbo codes, which need several repeated calculations at the receiver side. That is, in fact, the reason why a simple convolutional code was preferred in 3G voice transmission.

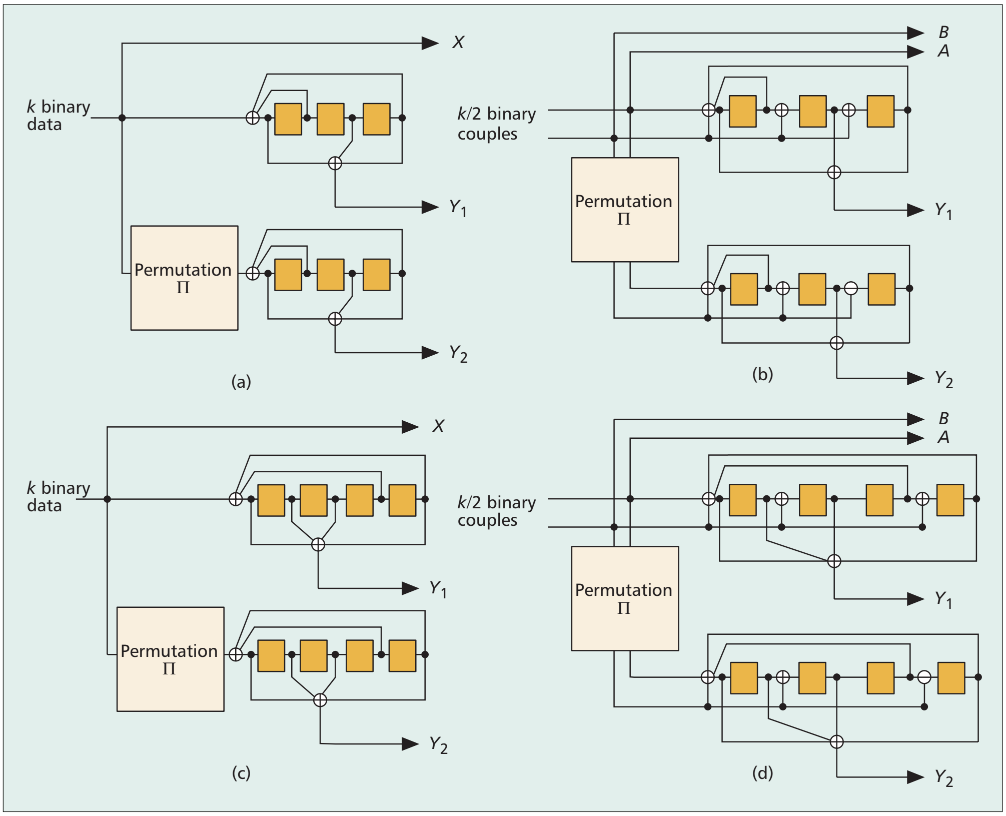

Figure 5. *The four turbo codes used in practice: a) 8-state binary;, b) 8-state duobinary, both with polynomials 15, 13 (or their symmetric form 13, 15); c) 16-state binary; d) 16-state duobinary, both with polynomials 23, 35 (or their symmetric form 31, 27). Binary codes are suitable for rates lower than 1/2, duobinary codes for rates higher than 1/2.*

be 2.2.16.*P*. The first coefficient 2 represents the number of component codes, and the second one is justified by the double, backward and forward, recursion. The number of states of each code accounts for coefficient 16, and finally, *P* is the number of iterations. Choosing $P = 6$ gives an equivalent number of states equal to 384. Assuming that computational complexity is proportional to the number of states, when using the same basic algorithm (MAP), the ratio would be around 5.10^6 for comparable performance! However, turbo decoding requires further operations like the calculation of extrinsic information and additional memory, and the gain would be slightly lower than this value, which still remains fairly impressive.

• Codes C and D are the two normalized channel codes for third-generation (3G) mobile telephony [14]. Code C, for voice transmission, is a simple 256-state convolutional code, decoded by the Viterbi algorithm. Code D, for data transmission, is a 2×8-state turbo code decoded by a simplified version of the MAP algorithm, called Max-Log-MAP. Assuming that the computational complexity of the latter is about twice that of the Viterbi decoder (again because of the double recursion), the equivalent number of states to be swept for the turbo code, with six iterations, is 192, that is, less than the number of states processed by the voice decoder! Of course, again, iterative decoding needs a certain amount of memory; moreover, the aimed for data rates are much higher. This explains why the implementation of the turbo decoder in 3G receivers is finally more complex than that of the Viterbi decoder.

To summarize these briefly presented examples, let us say that:

- If the aim is to approach closely the ideal performance, iterative decoding offers considerable savings, by several orders of magnitude, compared to a single code.
- The additional complexity turbo decoding demands, compared to the simple Viterbi decoder, abundantly used with 16 or 64 states over the last two decades, seems to be quite compatible with, and even below, what the progress of microelectronics has offered and continues to offer.

About Latency

The latency issue is a weak point for turbo codes, which need several repeated calculations at the receiver side. That is, in fact, the reason why a simple convolutional code was preferred in 3G voice transmission. For the time being, we just have to be patient and expect Moore's law to still be valid for some years to come. If so, higher circuit frequencies and larger possibilities of parallelism will reduce the latency effects to a negligible level for most applications. However, as for all block-oriented codes, the decoder has

Application	Turbo code	Termination	Polynomials	Rates
CCSDS (deep space)	Binary, 16-state	Tail bits	23, 33, 25, 37	1/6, 1/4, 1/3, 1/2
UMTS, cdma2000 (3G mobile)	Binary, 8-state	Tail bits	13, 15, 17	1/4, 1/3, 1/2
DVB-RCS (return channel over satellite)	Duobinary, 8-state	Circular	15, 13	1/3 up to 6/7
DVB-RCT (return channel over terrestrial)	Duobinary, 8-state	Circular	15, 13	1/2, 3/4
Inmarsat (M4)	Binary, 16-state	No	23, 35	1/2
Eutelsat (Skyplex)	Duobinary, 8-state	Circular	15, 13	4/5, 6/7

Table 1. *Current known applications of convolutional turbo codes.*

to wait for the entire encoded sequence to begin its process, and this is an unavoidable limitation.

About Versatility

A more and more sought-for quality in channel coding is versatility, that is, the possibility for the same coding principle and the same decoder to satisfy various needs in terms of coding rates and block sizes. There is probably no coding principle more versatile than convolutional coding, with respect to redundancy rate. By means of so-called *puncturing* and with the same encoder, the amount of redundancy can be adjusted with accuracy. Furthermore, for convolutional codes, there is also no termination principle more versatile than circular termination. For any size, a convolutional code can be considered as a circle (a perfect block code!), without any side effects or need for additional information. Thanks to these two very advantageous properties, turbo codes based on CRSC codes are, in our opinion and for the time being, the most flexible way to encode.

The Applications of Turbo Codes

Depending on the subject dealt with, error correcting codes can be divided into many families. From the point of view of the applications, we will consider here three domains:

- **Medium error rates** (corresponding roughly to 1 > FER > 10^{-4})

This is typically the domain of automatic repeat request (ARQ) systems and is also the more favorable level of error rates for turbo codes. To achieve near optimum performance, 8-state component codes are sufficient. Figure 5a depicts the practical binary turbo code used for these applications and coding rates equal to or lower than 1/2. For higher rates, the duobinary turbo code of Fig. 5b is preferable, because, for a given rate, less puncturing is required. For each of them, one example of performance, in FER as a function of E_b/N_0, is given in Fig. 6 (UMTS: $R = 1/3$, $k = 640$ and DVB-RCS: $R = 2/3$, $k = 1504$). Referring to the curves of Fig. 4, one can observe that a FER of 10^{-4} is obtained at about 0.9 and 0.6 dB, respectively, above the theoretical limits. This quasi-optimum performance is actually achieved with existing silicon decoders and can be reproduced for most coding rates and block sizes, even the shortest.

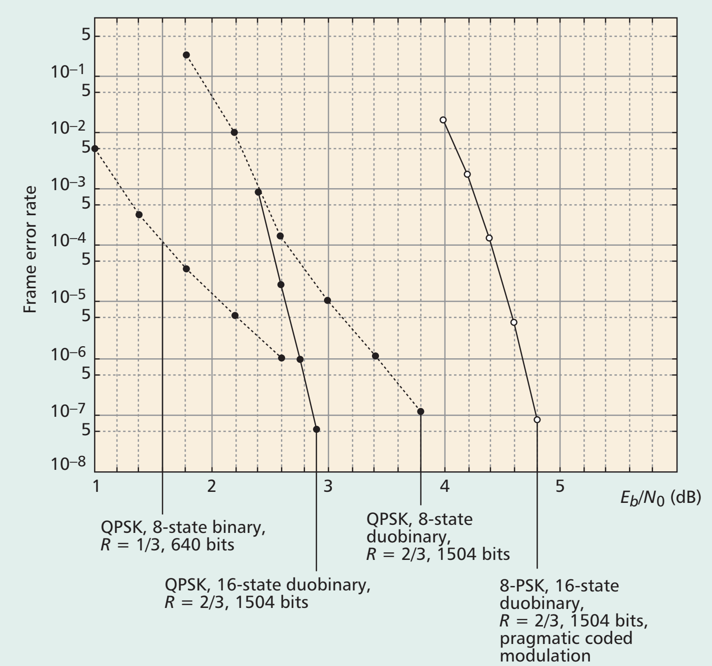

Figure 6. *Some examples of performance, expressed in FER, achievable with turbo codes on Gaussian channels. In all cases: decoding using the Max-Log-MAP algorithm with eight iterations and 4-bit(QPSK) or 5-bit (8-PSK) input quantization.*

- **Low error rates** (10^{-4} > FER > 10^{-9})

16-state turbo codecs perform better than 8-state ones, by about 1 dB for an FER of 10^{-7} (Fig. 6). Depending on the sought-for compromise between performance and decoding complexity, one can choose either one or the other. Figures 5c and 5d depict the 16-state turbo codes that can be used, binary for low rates, duobinary for high rates. In order to obtain good results at low error rates, the permutation function must be very carefully devised.

An example of performance, provided by the association of 8-PSK modulation and the turbo code of Fig. 5d, is also plotted in Fig. 6, for k = 1504 and a spectral efficiency of 2 b/s/Hz. This association is made according to the pragmatic approach (i.e., the codec is the same as that used for binary modulation). It just requires binary to octary conversion, at the transmitter side, and the converse at the receiver side. Again, the

Because turbo codes are in their adolescence, they are not yet able to answer all the requests, in terms of throughput, latency, and simplicity, and this still leaves some amount of work to be undertaken by the numerous researchers in the field.

result obtained with the actual constraints of the implementation is close (within 1 dB) to ideal performance.

• **Very low error rates** (10^{-9} > FER)

The largest minimum distances that can be obtained from turbo codes, for the time being, are not sufficient to prevent a slope change in the BER (E_b/N_0) or FER (E_b/N_0) curves at very low error rates. Compared to what is possible today, an increase of minimum distances by roughly 25 percent would be necessary to make turbo codes attractive for this type of application, such as optical transmission or mass storage error protection.

Table 1 summarizes the normalized applications of convolutional turbo codes known to date. The first three codes of Fig. 5 have been chosen for these various systems. The fourth one is under consideration for new satellite transmission systems. It is also likely that existing applications such as asynchronous digital subscriber line (ADSL) and LANs will choose one of the four codes of Fig. 5, or perhaps another not yet known turbo-like code, for their evolution.

Conclusion

The new challenges that have been faced in recent years, as well as those presently being addressed, for ever more powerful communications, require continuous innovation and, from time to time, some unconventional concepts. Among these, turbo codes have come at just the right time, benefiting from Moore's law and from the worldwide research effort to achieve Shannon's promises. Because turbo codes are in their adolescence, they are not yet able to answer all requests in terms of throughput, latency, and simplicity, and this still leaves some amount of work to be undertaken by the numerous researchers in the field. Also, beyond the simple introduction of a new error-correcting technique, the turbo principle (i.e., the way to process data in receivers so that no information is wasted) has opened up a new way of thinking in the construction of communication algorithms.

Acknowledgments

The author wishes to acknowledge the contributions of Janet Ormrod, Emeric Maury (for the curves of Fig. 4), Marc Fossorier, and the anonymous reviewers to the writing of this article.

References

[1] S.-Y Chung *et al.*, "On the Design of Low-Density Parity-Check Codes Within 0.0045 dB of the Shannon Limit," *IEEE Commun. Lett*, vol. 5, no. 2, Feb. 2001, pp. 58–60.

[2] J. Boutros *et al.*, "Turbo Code at 0.03 dB from Capacity Limit," *Proc. Int'l. Symp. Info. Theory*, June–July 2002, p. 56.

[3] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Tech. J.*, vol. 27, July and Oct. 1948.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-codes," *Proc. ICC '93*, Geneva, Switzerland, May 1993, pp. 1064–70.

[5] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-codes," *IEEE Trans. Commun.*, vol. 44, no. 10, Oct. 1996, pp. 1261–71.

[6] G. Battail, "Coding for the Gaussian Channel: the Promise of Weighted-Output Decoding," *Int'l J. Sat. Commun.*, vol. 7, 1989, pp. 183–92.

[7] J. Hagenauer and P. Hoeher, "A Viterbi Algorithm with Soft-Decision Outputs and Its Applications," *Proc. GLOBECOM '89*, Dallas, TX, Nov. 1989, pp. 47.11–47-17.

[8] P. Elias, "Error-free Coding," *IEEE Trans. Info. Theory*, vol. 4, no. 4, Sept. 1954, pp. 29–39.

[9] R. G. Gallager, "Low-Density Parity-Check Codes," *IRE Trans. Info. Theory*, vol. IT-8, Jan. 1962, pp. 21–28.

[10] R. M. Tanner, "A Recursive Approach to Low Complexity Codes," *IEEE Trans. Info. Theory*, vol. IT-27, Sept. 1981, pp. 533–47.

[11] G. D. Forney Jr., "Convolutional Codes I: Algebraic Structure," *IEEE Trans. Info. Theory*, vol. IT-16, no. 16, Nov. 1970, pp. 720–38.

[12] R. J. McEliece and D. J. C. MacKay, "Turbo Decoding as an Instance of Pearl's 'Belief Propagation' Algorithm," *IEEE JSAC*, vol. 16, no. 2, Feb. 1998, pp. 140–52.

[13] L. R. Bahl *et al.*, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. Info. Theory*, vol. IT-20, Mar. 1974, pp. 248–87.

[14] 3GPP Tech. Spec. Group, "Multiplexing and Channel Coding (FDD)," TS 25.212 v2.0.0, June 1999; TIA/EIA/IS 2000-2, "Physical Layer Standard for cdma2000 Spread Spectrum Systems," July 1999.

Biography

Claude Berrou (Claude.Berrou@enst-bretagne.fr) received an electrical engineering degree from the Institut National Polytechnique, Grenoble, France, in 1975. In 1978 he joined the ENST Bretagne, Brest, France, where he is currently a professor in the Electronics Department. His research topics are algorithm/silicon interaction, electronics and digital communications, error correcting codes, turbo codes, SISO decoders, genetic coding, and more. He is the author or co-author of eight registered patents and about 45 publications in the field of communications. He was co-recipient of the 1997 Stephen O. Rice Award for the best paper in *IEEE Transactions on Communications*, and the 1998 IEEE Information Theory Society Paper Award. He has also received several other distinctions, including the 1997 Médaille Ampère, one of the Golden Jubilee Awards for Technological Innovation (IEEE IT Society) in 1998, and the 2003 IEEE Richard W. Hamming Medal with Professor Glavieux.