The statistical mechanics of turbo codes.

Andrea Montanari
Scuola Normale Superiore and INFN – Sezione di Pisa
I-56100 Pisa, ITALIA
Internet: `montanar@cibs.sns.it`

Nicolas Sourlas
Laboratoire de Physique Théorique de l' Ecole Normale Supérieure [*]
24 rue Lhomond, 75231 Paris CEDEX 05, France.
Internet: `sourlas@lpt.ens.fr`

October 18, 2006

Abstract

The "turbo codes", recently proposed by Berrou et. al. [1] are written as a disordered spin Hamiltonian. It is shown that there is a threshold Θ such that for signal to noise ratios $v^2/w^2 > \Theta$ the error probability per bit vanishes in the thermodynamic limit, i.e. the limit of infinitly long sequences. The value of the threshold has been computed for two particular turbo codes. It is found that it depends on the code. These results are compared with numerical simulations.

---

[*]UMR 8549, Unité Mixte de Recherche du Centre National de la Recherche Scientifique et de l' Ecole Normale Supérieure.

1   Introduction.

The recent invention of "turbo codes" by Berrou and Glavieux [1] is considered a major breakthrough in communications. For the first time one can communicate almost error-free for signal to noise ratios very close to the theoretical bounds of information theory. Turbo codes are fastly becoming the new standard for error correcting codes in digital communications. The invention of turbo codes and their iterative decoding algorithm was empirical. There is no theoretical understanding of why they are so successfull. The decoding algorithm is thought to be an approximate algorithm. We think that turbo codes are interesting, even outside the context of communication theory, because they provide a non trivial example of a disordered system which can be studied numerically with a fast algorithm.

In this paper we will study turbo codes and turbo decoding using the modern tools of statistical mechanics of disordered systems. One of us has already shown in the past [4] that there is a mathematical equivalence between error correcting codes and theoretical models of spin glasses. In particular the logarithm of the probability for any given signal, conditional on the communication channel output, has the form of a spin glass Hamiltonian. We will construct the Hamiltonian which corresponds to the turbo codes and study its properties. This will clarify why they are so successfull. In particular we will show that there is a threshold Θ such that for signal to noise ratios $v^2/w^2 > \Theta$ the average error probability per bit $\overline{P_e}$ vanishes in the thermodynamic limit, i.e. the limit of infinitly long sequences. In $\overline{P_e}$ the average is taken over a large class of turbo codes (see later) and over "channel" noise. The rate of these codes is finite. The value of the threshold has been computed for two particular turbo codes. It was found that it depends on the code. We also compare these results with numerical simulations.

Our results are typical of the statistical mechanics approach: we study only the average performance of turbo codes, not the performance of any particular one. Furthermore there exist "very few" particular codes performing "much worse" than the average.

Let us first briefly remind the connection between error-correction codes and spin-glass models. In the mathematical theory of communication both the production of information and its transmission are considered as probabilistic events. A source is producing information messages according to a certain probability distribution. Messages of length N are sequences of N symbols or "letters of an alphabet" $a_1, a_2, \cdots, a_N$. We will assume for simplicity a binary alphabet, i.e. $a_i = 0$ or 1 and that all symbols are equally probable. Instead of a_i we can equally well use Ising spins

$$\sigma_i = (-1)^{a_i} = \pm 1 \tag{1.1}$$

The messages are sent through a noisy transmission channel. If a $\sigma = \pm 1$ is sent through the transmission channel, because of the noise, the output will be a real number σ^{out}, in general different from σ. Again, the statistical properties of the transmission channel are supposed to be known. Let us call $Q(\boldsymbol{\sigma}^{\text{out}}|\boldsymbol{\sigma})d\boldsymbol{\sigma}^{\text{out}}$ the probability for the transmission channel's output to be between $\boldsymbol{\sigma}^{\text{out}}$ and $\boldsymbol{\sigma}^{\text{out}}+d\boldsymbol{\sigma}^{\text{out}}$, when the input was $\boldsymbol{\sigma}$. $Q(\boldsymbol{\sigma}^{\text{out}}|\boldsymbol{\sigma})$ is supposed to be known. For reasons of simplicity, we assume that the noise is independent

for any pair of bits ("memoryless channel"), i.e.

$$Q(\boldsymbol{\sigma}^{\text{out}}|\boldsymbol{\sigma}) = \prod_i Q(\sigma_i^{\text{out}}|\sigma_i) \qquad (1.2)$$

In the case of a memoryless channel and a gaussian noise:

$$Q_{\text{gauss}}(\sigma^{\text{out}}|\sigma) \equiv \frac{1}{\sqrt{2\pi w^2}} \exp\left\{-\frac{(\sigma^{\text{out}} - \sigma)^2}{2w^2}\right\} \qquad (1.3)$$

Shannon calculated the channels capacity $\mathcal{C}$, i.e. the maximum information per use of the channel that can be transmitted.

$$\mathcal{C}_{\text{gauss}} = \frac{1}{2} \log_2(1 + \frac{v^2}{w^2}) \qquad (1.4)$$

where v^2 is the signal power.

Under the above assumptions, communication is a statistical inference problem. Given the transmission channel's output and the statistical properties of the source and of the channel, one has to infer what message was sent. In order to reduce communication errors, one may introduce (deterministic) redundancy into the message ("channel encoding") and use this redundancy to infer the message sent through the channel ("decoding"). The algorithms which transform the source outputs to redundant messages are called error-correcting codes. More precisely, instead of sending the N original bits σ_i, one sends M bits J_k^{in}, $k = 1, \cdots, M, M > N$, constructed in the following way

$$J_k^{\text{in}} = C_{i_1 \dots i_{l_k}}^{(k)} \sigma_{i_1} \cdots \sigma_{i_{l_k}} \qquad (1.5)$$

where the "connectivity" matrix $C_{i_1 \dots i_{l_k}}^{(k)}$ has elements zero or one. For any k, all the $C_{i_1 \dots i_{l_k}}^{(k)}$ except from one are equal to zero, i.e. the J_k^{in} are equal to ± 1. $C_{i_1 \dots i_{l_k}}^{(k)}$ defines the code, i.e. it tells from which of the σ's to construct the kth bit of the code.

This kind of codes are called parity checking codes because J_k^{in} counts the parity of the minusis among the l_k σ's. The ratio $R = N/M$ which specifies the reduancy of the code, is called the rate of the code.

Knowing the source probability, the noise probability, the code and the channel output, one has to infer the message that was sent. The quality of inference depends on the choice of the code.

According to the famous Shannon's channel encoding theorem, there exist codes which, in the limit of infinitly long messages, allow error-free communication, provided the rate of the code R is less than the channel capacity $\mathcal{C}$. This theorem says that such "ideal" codes exist, but does not say how to construct them.

We have shown that there exists a close mathematical relationship between error-correcting codes and theoretical models of disodered systems. As we previously said, the output of the channel is a sequence of M real numbers $\boldsymbol{J}^{\text{out}} = \{J_k^{\text{out}}, k = 1, \cdots, M\}$, which are random variables, obeying the probability distribution $Q(J_k^{\text{out}}|J_k^{\text{in}})$. Once the channel

output $\boldsymbol{J}^{\text{out}}$ is known, it is possible to compute the probability $P(\boldsymbol{\tau}|\boldsymbol{J}^{\text{out}})$ for any particular sequence $\boldsymbol{\tau} = \{\tau_i,\ i = 1, \cdots, N\}$ to be the *source* output (i.e. the information message).

More precisely, the equivalence between spin-glass models and error correcting codes is based on the following property.

The probability $P(\boldsymbol{\tau}|\boldsymbol{J}^{\text{out}})$ for any sequence $\boldsymbol{\tau}$ to be the information message, conditional on the channel output $\boldsymbol{J}^{\text{out}}$ is given by

$$\ln P(\boldsymbol{\tau}|\boldsymbol{J}^{\text{out}}) = \text{const} + \sum_{k=1}^{M} C^{(k)}_{i_1 \ldots i_{l_k}}\ B_k\ \tau_{i_1} \cdots \tau_{i_{l_k}} \equiv -H(\boldsymbol{\tau}) \tag{1.6}$$

where

$$B_k \equiv B(J_k^{\text{out}}) \equiv \frac{1}{2} \ln \frac{Q(J_k^{\text{out}}|1)}{Q(J_k^{\text{out}}|-1)} \tag{1.7}$$

We recognize in this expression the Hamiltonian of a p-spin spin-glass Hamiltonian. The distribution of the couplings is determined by the probability $Q(J^{\text{out}}|J^{\text{in}})$.

In the case when $Q(J^{\text{out}}|J^{\text{in}}) = Q(-J^{\text{out}}|-J^{\text{in}})$ (the case of a "symmetric channel"), $B(J^{\text{out}}) = -B_k(-J^{\text{out}})$ and one recovers the invariance of the spin-glass Hamiltonian under gauge transformations.

"Minimum error probability decoding" (or MED), which is widely used in communications [2], consists in choosing the most probable sequence $\boldsymbol{\tau}^0$. This is equivalent to finding the ground state of the above spin-glass Hamiltonian.

Instead of considering the most probable instance, one may only be interested in the most probable value τ_i^{MAP} of the "bit" τ_i (Maximum A posteriori Probability or MAP decoding) [3] which can be expressed in terms of the magnetization at temperature $T = 1/\beta$ equal to one [6]:

$$\tau_i^{\text{MAP}} = \text{sign}\ (m_i)\quad;\quad m_i = \frac{1}{Z} \sum_{\boldsymbol{\tau}} \tau_i\ \exp\{-H(\boldsymbol{\tau})\} \tag{1.8}$$

where $H(\boldsymbol{\tau})$ is defined by Eq.(1.6).

It is remarkable that $\beta = 1$ coincides with the Nishimori temperature in spin glasses [7]. MAP decoding is an essential ingredient in turbo decoding (see later).

When all messages are equally probable and the transmission channel is memoryless and symmetric, the error probability is the same for all input sequences. It is enough to compute it in the case where all input bits are equal to one. In this case, the error probability per bit P_e is

$$P_e = \frac{1}{2}(1 - m^{(d)}) \equiv \frac{1}{2} \left(1 - \frac{1}{N} \sum_{i=1}^{N} \tau_i^{(d)} \right) \tag{1.9}$$

and $\tau_i^{(d)}$ is the symbol sequence produced by the decoding procedure. One can derive from this a very general lower bound for P_e, using the analog of the low temperature expansion.

An obvious bound (for zero temperature decoding) is provided by the probability $P_e^{(1)}$ that only one bit is incorrect, i.e. $\tau_j = -1$ while all other bits are correct, i.e. $\tau_i = 1$ for all $i \neq j$:

$$P_e \geq P_e^{(1)} = \text{Probability of } \left\{ \sum_{k \in \Omega(j)} B_k < 0 \right\} \tag{1.10}$$

where the $\Omega(j)$ denotes the set of the couplings in which τ_j appears.
A necessary condition for transmitting without errors is that $\sum_{k \in \Omega(j)} B_k > 0$ with probability one. This is only possible if every spin appears in an infinite number of terms in the Hamiltonian. Let l_k be the number of spins coupled through the coupling B_k.
The total number of spins beeing N, a spin appears on the average in

$$\frac{1}{N} \sum_{k=1}^{M} l_k = \frac{M}{N} \frac{1}{M} \sum_{k=1}^{M} l_k = \frac{\bar{l}}{R} \tag{1.11}$$

terms, where $\bar{l}$ is the average of l_k (the number of spins coupled together) and R is the rate of the code.
So a necessary condition for a finite rate code to achieve zero error probability, is that the average number of spins coupled together diverges in the thermodynamic limit ($N \to \infty$). This condition is realised in Derrida's random energy model [5] which has been shown to be an ideal code [4] (in that case $R = 0$).
We will show in the following that this is also true for the case of recursive turbo codes, while it is not true for non recursive turbo codes.

2 Convolutional codes.

Convolutional codes are the building blocks of turbo codes. In this section we shall describe both non recursive and recursive convolutional codes and the corresponding spin models. The information message i.e. the source output (before encoding) will be denoted by:

$$\boldsymbol{\tau} \equiv (\tau_1, \ldots, \tau_N) \tag{2.1}$$

It is convenient to think of the source producing a symbol per unit time, i.e. in τ_i, i denotes the time. For simplicity we consider a code of rate $R = 1/2$. The encoded message has the form:

$$\boldsymbol{J} \equiv (J_1^{(1)}, \ldots, J_N^{(1)}; J_1^{(2)}, \ldots, J_N^{(2)}) \tag{2.2}$$

Any hardware implementation of a convolutional encoder contains a sequence of r memory registers. We shall call r the range of the code.
Let's denote by $\Sigma_1(t), \ldots, \Sigma_r(t)$ the content of the memory registers at time t. At each time step the content of each memory register is shifted to the right:

$$\Sigma_{j+1}(t+1) = \Sigma_j(t) \quad \text{for} \quad j = 1, \ldots, r-1 \tag{2.3}$$

Moreover for convenience of notation we define

$$\Sigma_0(t) \equiv \Sigma_1(t+1) \tag{2.4}$$

and the following sequence of bits which we shall call the register sequence:

$$\boldsymbol{\sigma} \equiv (\sigma_1, \ldots, \sigma_N), \quad \sigma_i \equiv \Sigma_0(i) \tag{2.5}$$

The sequence of the σ's is a function of the source sequence (which may depend on the code):

$$\boldsymbol{\tau} \quad \mapsto \quad \boldsymbol{\sigma}(\boldsymbol{\tau}) \equiv (\sigma_1(\boldsymbol{\tau}), \ldots, \sigma_N(\boldsymbol{\tau})) \tag{2.6}$$

When not ambiguous we shall omit in the following the functional dependence of $\boldsymbol{\sigma}$ upon $\boldsymbol{\tau}$.

For non recursive convolutional codes this application is extremely simple:

$$\sigma_i(\boldsymbol{\tau}) = \Sigma_0(i) = \tau_i \tag{2.7}$$

The encoded message $\boldsymbol{J}$ is easily defined in terms of the content of the register sequence:

$$J_i^{(n)} = \prod_{j=0}^{r}(\Sigma_j(i))^{\kappa(j;n)} = \prod_{j=0}^{r}(\sigma_{i-j})^{\kappa(j;n)} \tag{2.8}$$

$$i = 1, \ldots, N \; ; \; n = 1, 2$$

$$\kappa(j;n) \in \{0, 1\}$$

We shall assume hereafter that $\kappa(0;1) = \kappa(0;2) = 1$.

To avoid redundancy we choose r such that either $\kappa(r;1)$ or $\kappa(r;2)$ are different from 0. To make Eq.(2.8) meaningful for $i = 1, \ldots, r$ we define $\sigma_j = +1$ for $j \leq 0$. Notice however that the exact definition of $J_1^{(n)}, \ldots, J_r^{(n)}$ is irrelevant in the thermodynamic limit.

The numbers $\kappa(j;n)$ define the code. Several conventions are used to give them in a compact form. A simple and useful one is the following. To each of the two sets of numbers $\kappa(j;1)$ and $\kappa(j;2)$ is associated a polynomial on $\mathbb{Z}_2$

$$g_n(x) = \sum_{j=0}^{r} \kappa(j;n)x^j. \tag{2.9}$$

The g_n are called generating polynomials. In the same way we can associate a polynomial to the register sequence ($G(x) \equiv \sum_{j=1}^{N} b_j x^j$; $\sigma_j \equiv (-1)^{b_j}$), to the source message ($H(x) \equiv \sum_{j=1}^{N} a_j x^j$; $\tau_j \equiv (-1)^{a_j}$) and to each part of the encoded message ($\mathcal{G}^{(n)}(x) \equiv \sum_{j=1}^{N} d_j^{(n)} x^j$; $J_j^{(n)} \equiv (-1)^{d_j^{(n)}}$). With these definitions it is evident that the correspondence (2.7) between the source and the register sequences for a non recursive convolutional code implies:

$$G(x) = H(x) \tag{2.10}$$

and the encoding rule (2.8) becomes

$$\mathcal{G}^{(n)}(x) = g_n(x)G(x) = g_n(x)H(x) \tag{2.11}$$

A few examples are the following:

(a). The simplest non trivial convolutional code has range 1:

$$J_i^{(1)} = \sigma_i \sigma_{i-1} \quad \Rightarrow \quad g_1(x) = 1 + x \tag{2.12}$$

$$J_i^{(2)} = \sigma_i \quad \Rightarrow \quad g_2(x) = 1 \tag{2.13}$$

(b). A simple code with range $r = 2$ whose behavior will be examined in what follows:

$$J_i^{(1)} = \sigma_i \sigma_{i-1} \sigma_{i-2} \quad \Rightarrow \quad g_1(x) = 1 + x + x^2 \tag{2.14}$$

$$J_i^{(2)} = \sigma_i \sigma_{i-2} \quad \Rightarrow \quad g_2(x) = 1 + x^2 \tag{2.15}$$

(c). The code with range $r = 4$ used by Berrou and collaborators to build the first example of turbo code:

$$J_i^{(1)} = \sigma_i \sigma_{i-1} \sigma_{i-2} \sigma_{i-3} \sigma_{i-4} \quad \Rightarrow \quad g_1(x) = 1 + x + x^2 + x^3 + x^4 \tag{2.16}$$

$$J_i^{(2)} = \sigma_i \sigma_{i-4} \quad \Rightarrow \quad g_2(x) = 1 + x^4 \tag{2.17}$$

Recursive convolutional codes are most easily defined in terms of the generating polynomials. The difference with non recursive codes is in the relation between the source and the register sequences. In the non recursive case it was given by Eq.(2.7) or by Eq.(2.10). In the recursive case one has:

$$G(x) = \frac{1}{g_1(x)} H(x) \tag{2.18}$$

so that Eq. (2.11) gives

$$\mathcal{G}^{(1)}(x) = g_1(x) G(x) = H(x) \;, \quad \mathcal{G}^{(2)}(x) = g_2(x) G(x) = \frac{g_2(x)}{g_1(x)} H(x) \tag{2.19}$$

Two different recursive codes can be defined by permuting the two polynomials g_1 and g_2. It is easy to show that Eq. (2.19) is equivalent to

$$\sigma_i(\boldsymbol{\tau}) = \Sigma_0(i) = \tau_i \prod_{j=1}^{r} \Sigma_j(i)^{\kappa(j;1)} \;=\; \tau_i \prod_{j=1}^{r} (\sigma_{i-j})^{\kappa(j;1)} \tag{2.20}$$

From Eq.(2.20) it follows that:

$$\tau_i(\boldsymbol{\sigma}) = \prod_{j=0}^{r} (\sigma_{i-j})^{\kappa(j;1)} \;=\; J_i^{(1)} \tag{2.21}$$

Because of the last equality in Eq.(2.21) a part of the encoded message (in the recursive case) is always the message itself.

We shall now consider decoding. Using the method explained in the Introduction the probability distribution of the register sequence conditional to some ouput can be written as the Boltzmann weight of a spin model with random couplings. The Hamiltonian of this model is:

$$H(\boldsymbol{\sigma}; \boldsymbol{J}^{\text{out}}) = -\sum_{i=1}^{N} \left\{ B(J_i^{(1),\text{out}}) \prod_{j=0}^{r} (\sigma_{i-j})^{\kappa(j;1)} + B(J_i^{(2),\text{out}}) \prod_{j=0}^{r} (\sigma_{i-j})^{\kappa(j;2)} \right\}$$

(2.22)

where $B(\cdot)$ is defined in Eq.(1.7).

For convolutional codes the model is one dimensional with two types of couplings. The range of the interaction coincide with the range of the code. The alert reader will notice that the Hamiltonian is expressed as a function of the spins of the register sequence σ_i, instead of the source sequence τ_i used in the introduction. For non recursive codes $\sigma_i = \tau_i$. For recursive codes τ_i is given by Eq.(2.21), i.e. in this last case decoding can be thought of as the computation of an expectation value of a composite operator. However the spin Hamiltonian is the same for both the recursive and not recursive codes.

We define the decoding at arbitrary temperature $T \equiv 1/\beta$ as follows:

$$\tau_i^{\beta} \equiv \text{sign}(\langle \tau_i(\boldsymbol{\sigma}) \rangle_{\beta})$$

(2.23)

$$\langle O(\boldsymbol{\sigma}) \rangle_{\beta} \equiv \frac{1}{Z(\boldsymbol{J}^{\text{out}}; \beta)} \sum_{\boldsymbol{\sigma}} O(\boldsymbol{\sigma}) \exp\{-\beta H(\boldsymbol{\sigma}; \boldsymbol{J}^{\text{out}})\}$$

(2.24)

where the expression for $\tau_i(\boldsymbol{\sigma})$ is given by Eq.(2.21) or by Eq.(2.7) depending whether the code is recursive or not.

As seen in the introduction there are two widely used decoding strategies:

- Maximum Likelihood decoding which consists in finding the most probable sequence of bits and corresponds to the choice $\beta = \infty$ in Eq.(2.23): $\tau_i^{ML} \equiv \tau_i^{\beta=\infty}$.

- Maximum A posteriori Probability decoding which consists in finding the most probable sequence of bits and corresponds to the choice $\beta = 1$ in Eq.(2.23): $\tau_i^{MAP} \equiv \tau_i^{\beta=1}$. This is the strategy which enters in turbo decoding.

Both this strategies can be implemented in a very efficient way using the transfer matrix technique. The corresponding algorithms are known in communication theory as the Viterbi algorithm [2] for the $\beta = \infty$ case and the BCJR algorithm [3] for the $\beta = 1$ case. The complexity of these algorithms grows like $N2^r$.

The use of the register sequence (i.e. of the σ variables) makes evident the similarity between recursive and nonrecursive codes: they correspond to the same spin model. This implies e.g. that, if zero temperature decoding is adopted, the probability of transmitting a message without errors is the same with the two codes.

In the limit $r \to \infty$ it is possible to construct convolutional codes corresponding to spin models with infinite connectivity and couplings between an infinite number of spins. They should allow to transmit without errors when the noise is low enough. In practice, because of the growing complexity of the transfer matrix algorithm, a compromise between low

r's (which are simpler to decode) and high r's (which show better performances) must be found. The values of r used in practical cases are between 2 and 7.

We can write the decoding strategy in terms of the message (i.e. of the τ variables) without making use of the register sequence (i.e. of the σ variables):

$$\tau_i^\beta = \text{sign}\left\{\frac{1}{Z(\boldsymbol{J}^{\text{out}};\beta)}\sum_{\boldsymbol{\tau}}\tau_i\exp\{-\beta H(\boldsymbol{\sigma}(\boldsymbol{\tau});\boldsymbol{J}^{\text{out}})\}\right\} \tag{2.25}$$

For non recursive codes, because of Eq.(2.7), things remain unchanged. However, for recursive codes, since Eq.(2.21) cannot be inverted in a local way, we obtain a non local Hamiltonian.

As a simple illustration of this observation we can consider the Hamiltonian corresponding to the code (a):

$$H^{(\text{a})}(\boldsymbol{\sigma};\boldsymbol{J}) = -\sum_{i=1}^{N}B(J_i^{(1),\text{out}})\sigma_i\sigma_{i-1} - \sum_{i=1}^{N}B(J_i^{(2),\text{out}})\sigma_i \tag{2.26}$$

$$H^{(\text{a})}(\boldsymbol{\sigma}(\boldsymbol{\tau});\boldsymbol{J}) = -\sum_{i=1}^{N}B(J_i^{(1),\text{out}})\tau_i - \sum_{i=1}^{N}B(J_i^{(2),\text{out}})\prod_{j=1}^{i}\tau_j \tag{2.27}$$

For less simple codes we define the numbers $\rho(j)\in\{0,1\}$ as follows:

$$\frac{g_2(x)}{g_1(x)} \equiv \sum_{j=0}^{\infty}\rho(j)x^j \pmod{2} \tag{2.28}$$

we get

$$H(\boldsymbol{\sigma}(\boldsymbol{\tau});\boldsymbol{J}) = -\sum_{i=1}^{N}B(J_i^{(1),\text{out}})\tau_i - \sum_{i=1}^{N}B(J_i^{(2),\text{out}})\prod_{j=0}^{i-1}(\tau_{i-j})^{\rho(j)} \tag{2.29}$$

Written in this form recursive codes look very different from non recursive ones with the same range. If $g_2(x)$ is not divisible by $g_1(x)$ the corresponding spin models have infinite connectivity and interactions with infinite range; they are similar, in this respect, to $r=\infty$ non recursive codes.

Neverthless they do not behave, in general, radically better than the non recursive codes with the same range because there exists, as we have shown, a change of variables (from τ to σ) which makes the model local.

3   Turbo codes.

A turbo code is defined by the choice of a convolutional code and of a permutation of N objects. We use for the permutation the following notation:

$$P:\{1,\ldots,N\} \quad \to \quad \{1,\ldots,N\} \tag{3.1}$$
$$i \quad \mapsto \quad P(i) \tag{3.2}$$

and we shall denote by P^{-1} the inverse permutation $(P(P^{-1}(i)) = P^{-1}(P(i)) = i)$.

The basic idea is to apply the permution P to the source sequence $\boldsymbol{\tau}$ to produce a new sequence $\boldsymbol{\tau}^P$. Obviously $\boldsymbol{\tau}^P$ does not carry any new information because P is known. Both sequences $\boldsymbol{\tau}$ and $\boldsymbol{\tau}^P$ are the inputs to two set of registers, each one implementing a convolutional encoding. In this way the rate of the code is decreased (i.e. greater redundancy). One can increase the rate by erasing some of the outputs [1], but we will not consider this possibility in this paper.

The properties of the system can strongly depend on the choice of the permutation. Permutations "near" the identity give very bad codes. We shall think to a "good" permutation as to a random permutation. In the limit $N \to \infty$ they are "far" from the identity with probability one. We shall discuss this point later in this section.

We illustrate this idea with the example of a rate $1/2$ recursive convolutional code, defined by the constants $\kappa(j;1)$ and $\kappa(j;2)$. The two register sequences are:

$$\sigma_i^{(1)} \equiv \sigma_i(\boldsymbol{\tau}) \quad \Rightarrow \quad \tau_i = \prod_{j=1}^{r}(\sigma_{i-j}^{(1)})^{\kappa(j;1)} \equiv \epsilon_i(\boldsymbol{\sigma}^{(1)}) \tag{3.3}$$

$$\sigma_i^{(2)} \equiv \sigma_i(\boldsymbol{\tau}^P) \quad \Rightarrow \quad \tau_i^P = \prod_{j=1}^{r}(\sigma_{i-j}^{(2)})^{\kappa(j;1)} \equiv \epsilon_i(\boldsymbol{\sigma}^{(2)}) \tag{3.4}$$

where $\boldsymbol{\tau}^P$ is the permuted message $(\tau_i^P \equiv \tau_{P(i)})$.

The relation between the two register sequences is rather involved and nonlocal for a general choice of the permutation. Moreover $\sigma_i^{(1)}$ can be expressed only in terms of a large number of $\sigma_j^{(2)}$'s. The identity permutation is clearly an exception since in this case $\sigma_i^{(1)} = \sigma_i^{(2)}$.

Let us consider as an example the code (a):

$$\sigma_i^{(1)} = \prod_{j=1}^{i} \sigma_{P(j)}^{(2)} \sigma_{P(j)-1}^{(2)} \tag{3.5}$$

It is simple to show that, for a random permutation, the number of different $\sigma^{(2)}$'s in the product on the r.h.s. of Eq.(3.5) is of order $O(N)$.

The turbo code defined by the permutation P and by the numbers $\kappa(j;1)$ and $\kappa(j;2)$ has rate $1/3$ and the encoded message has the following form:

$$J_i^{(0)} \equiv \prod_{j=0}^{r}(\sigma_{i-j}(\boldsymbol{\tau}))^{\kappa(j;1)} = \tau_i \tag{3.6}$$

$$J_i^{(1)} \equiv \prod_{j=0}^{r}(\sigma_{i-j}(\boldsymbol{\tau}))^{\kappa(j;2)} \tag{3.7}$$

$$J_i^{(2)} \equiv \prod_{j=0}^{r}(\sigma_{i-j}(\boldsymbol{\tau}^P))^{\kappa(j;2)} \tag{3.8}$$

It turns out that it is convenient to write the corresponding Hamiltonian as a function of both register sequences. This introduces new degrees of freedom and the Hamiltonian

is a function of $2N$ instead of N spin. The unwanted degrees of freedom are eliminated by imposing the constraint $\tau_i^P = \tau_{P(i)}$. This constraint can be written in terms of the σ's using Eqs.(3.3) and (3.4). The probability distribution for the register sequences can then be written as:

$$
P(\boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)} | \boldsymbol{J}^{\text{out}}) \;=\; \frac{1}{Z(\boldsymbol{J}^{\text{out}})} \prod_{i=1}^{N} \delta(\epsilon_{P(i)}(\boldsymbol{\sigma}^{(1)}), \epsilon_i(\boldsymbol{\sigma}^{(2)}))
$$
$$
\exp\{-H(\boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}; \boldsymbol{J}^{\text{out}})\} \tag{3.9}
$$
$$
H(\boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}; \boldsymbol{J}^{\text{out}}) \;=\; -\sum_{i=1}^{N} \left\{ B(J_i^{(0),\text{out}}) \prod_{j=0}^{r} (\sigma_{i-j}^{(1)})^{\kappa(j;1)} + \right.
$$
$$
+ B(J_i^{(1),\text{out}}) \prod_{j=0}^{r} (\sigma_{i-j}^{(1)})^{\kappa(j;2)} + \tag{3.10}
$$
$$
\left. + B(J_i^{(2),\text{out}}) \prod_{j=0}^{r} (\sigma_{i-j}^{(2)})^{\kappa(j;2)} \right\}
$$

where $\delta(i,j)$ is the ordinary Kronecker function. In this way the probability distribution is a local function of the spin variables $\sigma^{(1)}$ and $\sigma^{(2)}$.

We shall call the code defined by Eqs.(3.6,3.7,3.8) a non recursive turbo code if $\kappa(j;1) = \delta_{j,0}$ and a recursive turbo code otherwise. Recursive turbo codes are the ones usually called turbo codes in communication theory.

The probability distribution for the recursive turbo code (3.9) can't be written in terms of one of the two register sequences $\boldsymbol{\sigma}^{(1)}$ or $\boldsymbol{\sigma}^{(2)}$ without producing large connectivities (see Eq.(3.5)).

If P is the identity permutation then $\boldsymbol{\sigma}^{(1)} = \boldsymbol{\sigma}^{(2)}$ and the code becomes a convolutional one with the same rate $(1/3)$ and the same generating polynomials. We shall use the convolutional code obtained in this way as a standard comparison term for the performances of turbo codes (see Figs.(4-6)). The outcome of this comparison (i.e. recursive turbo codes have a much lower error probability than convolutional codes) demonstrates the importance of the choice of the permutation.

For non recursive turbo codes the two register sequences are related simply by a permutation:

$$
\sigma_i^{(1)} = \sigma_i(\boldsymbol{\tau}) = \sigma_{P^{-1}(i)}(\boldsymbol{\tau}^P) = \sigma_{P^{-1}(i)}^{(2)} \tag{3.11}
$$

and

$$
P_{\text{non-rec}}(\boldsymbol{\sigma}^{(1)} | \boldsymbol{J}^{\text{out}}) \;=\; \frac{1}{Z(\boldsymbol{J}^{\text{out}})} \exp\{-H(\boldsymbol{\sigma}^{(1)}, (\boldsymbol{\sigma}^{(1)})^P; \boldsymbol{J}^{\text{out}})\} \tag{3.12}
$$
$$
H(\boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}; \boldsymbol{J}^{\text{out}}) \;\equiv\; -\sum_{i=1}^{N} \left\{ B(J_i^{(0)\text{out}}) \sigma_i^{(1)} + B(J_i^{(1)\text{out}}) \prod_{j=0}^{r} (\sigma_{i-j}^{(1)})^{\kappa(j;2)} + \right.
$$
$$
\left. B(J_i^{(2)\text{out}}) \prod_{j=0}^{r} (\sigma_{i-j}^{(2)})^{\kappa(j;2)} \right\} \tag{3.13}
$$

11

so that the spin model corresponding to this type of code has a finite connectivity $c = 1 + 2\sum_{j=0}^{r}\kappa(j;2)$.

This finite versus infinite connectivity is the essential difference between non recursive and recursive turbo codes and explains why recursive turbo codes are so better and why they can achieve zero error probability for low enough noise.

We now discuss decoding. There is no exact decoding algorithm for turbo codes. Berrou et al. have proposed a very ingenious algorithm, called turbo decoding, which is thought to be approximate. Turbo decoding is an iterative procedure. At each step of the iteration, one considers one of the two chains, i.e. either the couplings $\boldsymbol{J}^{(0)}$ and $\boldsymbol{J}^{(1)}$ or $\boldsymbol{J}^{(0)}$ and $\boldsymbol{J}^{(2)}$ and proceeds to MAP decoding. The information so obtained is injected to the next step by adding appropriate external fields to the Hamiltonian. The algorithm terminates if a fixed point is reached.

In order to explain the algorithm more precisely, we introduce the following expectation values:

$$\Xi_i[\boldsymbol{B},\boldsymbol{B}'] \equiv \frac{1}{Z}\sum_{\boldsymbol{\sigma}}\epsilon_i(\boldsymbol{\sigma})\exp\left\{\sum_{i=1}^{N}B_i\prod_{j=0}^{r}(\sigma_{i-j})^{\kappa(j;1)} + \sum_{i=1}^{N}B_i'\prod_{j=0}^{r}(\sigma_{i-j})^{\kappa(j;2)}\right\} \quad (3.14)$$

The Ξ_i's can be computed in an efficient way by using the finite temperature transfer matrix algorithm. They are the expectation values of the operator defined by Eqs.(3.3) or (3.4).

Then we introduce the iteration variables:

$$\boldsymbol{\theta}^{(m)}(t) \equiv (\theta_1^{(m)}(t),\ldots,\theta_N^{(m)}(t)) \quad (3.15)$$
$$\boldsymbol{\Gamma}^{(m)}(t) \equiv (\Gamma_1^{(m)}(t),\ldots,\Gamma_N^{(m)}(t)) \quad (3.16)$$

for $m = 1,2$.

In terms of these variables the iteration reads

$$\theta_i^{(1)}(t+1) = \Xi_i[\boldsymbol{B}^{(0)} + \boldsymbol{\Gamma}^{(1)}(t),\boldsymbol{B}^{(1)}] \quad (3.17)$$
$$\theta_i^{(2)}(t+1) = \Xi_i[\boldsymbol{B}^{(0),P} + \boldsymbol{\Gamma}^{(2)}(t),\boldsymbol{B}^{(2)}] \quad (3.18)$$
$$\Gamma_i^{(1)}(t+1) = \operatorname{arctanh}\left[\theta_{P^{-1}(i)}^{(2)}(t+1)\right] - \Gamma_{P^{-1}(i)}^{(2)}(t) - B_i^{(0)} \quad (3.19)$$
$$\Gamma_i^{(2)}(t+1) = \operatorname{arctanh}\left[\theta_{P(i)}^{(1)}(t+1)\right] - \Gamma_{P(i)}^{(1)}(t) - B_{P(i)}^{(0)} \quad (3.20)$$

with

$$\boldsymbol{B}^{(m)} \equiv (B(J_1^{(m)\text{out}}),\ldots,B(J_N^{(m)\text{out}})) \ ; \ m = 0,1,2 \quad (3.21)$$

and $B_i^{(0),P} \equiv B_{P(i)}^{(0)}$.

The meaning of the previous equations is the following. The θ_i are expectation values of a sequence of operators which can take only values ± 1, computed independently for every element of the sequence. The information contained in θ_i can therefore be represented by an "external field" Γ_i such that $\theta_i = \tanh\Gamma_i$. In order to avoid double counting of information one substracts the external fields of the previous iteration as shown in

Eqs.(3.19,3.20).

Hopefully the iteration converges to a fixed point:

$$\lim_{t \to \infty} \theta_i^{(1)}(t) = \lim_{t \to \infty} \theta_{P^{-1}(i)}^{(2)}(t) \equiv \theta_i^* \tag{3.22}$$

The decoded message is obtained as follows:

$$\tau_i^{TURBO} \equiv \text{sign}(\theta_i^*) \tag{3.23}$$

The system described by Eq.(3.9,3.10) is seen in turbo decoding as the union of two one dimensional subsystem. Each subsystem acts on the other one through a magnetic field (in the non recursive case) or through an additional coupling (in the recursive case). To get some insight of Eqs.(3.17-3.20) we define the free energy functionals $F^{(1)}$ and $F^{(2)}$:

$$\mathcal{Z}^{(1)}[\mathbf{\Gamma}] = \sum_{\boldsymbol{\sigma}} \exp \left\{ \sum_{i=1}^{N} (B(J_i^{(0)}) + \Gamma_i) \prod_{j=0}^{r} (\sigma_{i-j})^{\kappa(j;1)} + \sum_{i=1}^{N} B(J_i^{(1)}) \prod_{j=0}^{r} (\sigma_{i-j})^{\kappa(j;2)} \right\} \tag{3.24}$$

$$\mathcal{Z}^{(2)}[\mathbf{\Gamma}] = \sum_{\boldsymbol{\sigma}} \exp \left\{ \sum_{i=1}^{N} (B(J_{P(i)}^{(0)}) + \Gamma_{P(i)}) \prod_{j=0}^{r} (\sigma_{i-j})^{\kappa(j;1)} + \sum_{i=1}^{N} B(J_i^{(2)}) \prod_{j=0}^{r} (\sigma_{i-j})^{\kappa(j;2)} \right\} \tag{3.25}$$

$$F^{(m)}[\boldsymbol{\theta}] \equiv \left. \sum_{i=1}^{N} \theta_i \Gamma_i - \log \left(\mathcal{Z}^{(m)}[\mathbf{\Gamma}] \right) \right|_{\theta_i = \frac{\partial \log(\mathcal{Z}^{(m)})}{\partial \Gamma_i}} \tag{3.26}$$

It is then simple to show that θ^* is a solution of the equation:

$$\frac{\partial}{\partial \theta_i} \mathcal{F}^{\text{turbo}}[\boldsymbol{\theta}] = 0 \tag{3.27}$$

where

$$\mathcal{F}^{\text{turbo}}[\boldsymbol{\theta}] \equiv F^{(1)}[\boldsymbol{\theta}] + F^{(2)}[\boldsymbol{\theta}] - F_0[\boldsymbol{\theta}] \tag{3.28}$$

$$F_0[\boldsymbol{\theta}] \equiv \sum_{i=1}^{N} \left\{ -B_i^{(0)} \theta_i - s(\theta_i) \right\} \tag{3.29}$$

$$s(x) \equiv - \left(\frac{1+x}{2} \right) \log \left(\frac{1+x}{2} \right) - \left(\frac{1-x}{2} \right) \log \left(\frac{1-x}{2} \right) \tag{3.30}$$

Eq.(3.28) is an approximation to the true free energy functional of the total system which is given by:

$$\mathcal{Z}[\mathbf{\Gamma}] \equiv \sum_{\boldsymbol{\sigma}^{(1)}} \sum_{\boldsymbol{\sigma}^{(2)}} \prod_{i=1}^{N} \delta(\epsilon_{P(i)}(\boldsymbol{\sigma}^{(1)}), \epsilon_i(\boldsymbol{\sigma}^{(2)}))$$

$$\exp \left\{ -H(\boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}; \boldsymbol{J}^{\text{out}}) + \sum_{i=1}^{N} \Gamma_i \epsilon_i(\boldsymbol{\sigma}^{(1)}) \right\} \tag{3.31}$$

$$\mathcal{F}[\boldsymbol{\theta}; \boldsymbol{J}^{(0)}, \boldsymbol{J}^{(1)}, \boldsymbol{J}^{(2)}] \equiv \left. \sum_{i=1}^{N} \theta_i \Gamma_i - \log \left(\mathcal{Z}[\mathbf{\Gamma}] \right) \right|_{\theta_i = \frac{\partial \log(\mathcal{Z})}{\partial \Gamma_i}} \tag{3.32}$$

13

where $H(\boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}; \boldsymbol{J}^{\text{out}})$ is given in Eq.(3.10).

It is then evident that

$$\mathcal{F}^{\text{turbo}}[\boldsymbol{\theta}] = \mathcal{F}[\boldsymbol{\theta}; \boldsymbol{J}^{(0)}, \boldsymbol{J}^{(1)}, \boldsymbol{0}] + \mathcal{F}[\boldsymbol{\theta}; \boldsymbol{J}^{(0)}, \boldsymbol{0}, \boldsymbol{J}^{(2)}] - \mathcal{F}[\boldsymbol{\theta}; \boldsymbol{J}^{(0)}, \boldsymbol{0}, \boldsymbol{0}] \qquad (3.33)$$

i.e. turbo decoding neglects terms of order $B(J_{i_1}^{(1)})B(J_{i_2}^{(2)})$.

4   Replica approach.

We would like to compute the error probability per bit. As explained in the introduction, in the case of a symmetric transmission channel, it is enough to compute the magnetization in the case of all inputs $\tau_i = 1$. The error probability per bit is given by the probability of a local magnetization being negative.

The similarity of the Hamiltonian (3.10) with the Hamiltonians of disordered spin systems is obvious. The disorder in the case of turbo codes has two origins. One is due to the (random) permutation which defines the particular code. The other is more conventional and is related to the randomness of the couplings which is due to the transmission noise. As usual in disordered systems, we can only compute the average over disorder and for that we have to introduce replicas.

Let us define the expectation value of the operator $\tau_i(\boldsymbol{\sigma})$ defined in Eqs.(3.3,3.4) with respect to the probability distribution given by Eqs.(3.9,3.10):

$$\Theta_i[\boldsymbol{J}^{\text{out}}, P] \equiv \sum_{\boldsymbol{\sigma}^{(1)}} \sum_{\boldsymbol{\sigma}^{(2)}} \epsilon_i(\boldsymbol{\sigma}^{(1)}) P(\boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)} | \boldsymbol{J}^{\text{out}}) \qquad (4.1)$$

The statistical properties of a turbo code can be derived from the probability distribution of this expectation value:

$$\mathcal{P}_i(\theta|P) \equiv \int dQ[\boldsymbol{J}^{\text{out}}] \, \delta\left(\theta - \Theta_i[\boldsymbol{J}^{\text{out}}, P]\right) \quad i = 1, \dots, N \qquad (4.2)$$

where

$$dQ[\boldsymbol{J}^{\text{out}}] = \prod_{n=0}^{2} \prod_{i=1}^{N} Q(J_i^{(n)}| + 1) dJ_i^{(n)} \qquad (4.3)$$

Then we define the average distribution

$$\overline{\mathcal{P}}(\theta) \equiv \frac{1}{N!} \sum_{P} \int dQ[\boldsymbol{J}^{\text{out}}] \, \delta\left(\theta - \Theta_i[\boldsymbol{J}^{\text{out}}, P]\right) \qquad (4.4)$$

where the sum runs over all possible permutations. $\overline{\mathcal{P}}(\theta)$ is expected not to depend upon the site i in the thermodynamic limit ($N \to \infty$).

The average error probability per bit is given by

$$\overline{P_e} \equiv \int_{-\infty}^{0} d\theta \, \overline{\mathcal{P}}(\theta) \qquad (4.5)$$

14

In any case $\overline{P_e}$ is an upper bound for the error probability of the "best" code (i.e. the one buildt with the permutation which yields the lowest error probability).

The replicated partition function is given by:

$$\overline{Z^n} \equiv \frac{1}{N!} \sum_P \int dQ[\boldsymbol{J}^{\text{out}}] \sum_{\{\boldsymbol{\sigma}^{(1),a}\}} \sum_{\{\boldsymbol{\sigma}^{(2),a}\}} \prod_{a=1}^n \prod_{i=1}^N \delta(\epsilon_{P(i)}(\boldsymbol{\sigma}^{(1),a}), \epsilon_i(\boldsymbol{\sigma}^{(2),a}))$$
$$\exp\left\{ -\sum_{a=1}^n H(\boldsymbol{\sigma}^{(1),a}, \boldsymbol{\sigma}^{(2),a}; \boldsymbol{J}^{\text{out}}) \right\} \tag{4.6}$$

The average over permutations can be done by introducing a matrix representation of the permutation

$$C_{ij}^P \equiv \delta_{i,P(j)} \quad ; \quad i,j = 1, \ldots, N \tag{4.7}$$

To sum over permutations, one sums over all matrices $C_{ij}^P = 0$ or 1 with the constrain $\sum_i C_{ij}^P = \sum_i C_{ij}^P = 1$. One may use the identity $\delta(\sigma, \tau) = (1 + \sigma\tau)/2$ to write

$$\prod_{a=1}^n \prod_{i=1}^N \delta(\epsilon_{P(i)}(\boldsymbol{\sigma}^{(1),a}), \epsilon_i(\boldsymbol{\sigma}^{(2),a})) = \prod_{a=1}^n \prod_{i,j=1}^N \left[\left(1 - \frac{1}{2}C_{ij}^P \right) + \frac{1}{2}C_{ij}^P \epsilon_i(\boldsymbol{\sigma}^{(1),a})\epsilon_j(\boldsymbol{\sigma}^{(2),a}) \right] \tag{4.8}$$

It can be shown [9] that the "effective action" which is obtained in this way describes two one-dimensional models coupled by a mean field-like interaction.

This is easily seen by making use of the occupation densities [8] defined below:

$$c_m(\underline{\epsilon}) \equiv \frac{1}{N} \sum_{i=1}^N \delta_{\underline{\epsilon}, \underline{\epsilon}_i^{(m)}} \tag{4.9}$$
$$\underline{\epsilon} \equiv (\epsilon^1, \ldots, \epsilon^a, \ldots, \epsilon^n) \in \{-1, +1\}^n$$
$$\underline{\epsilon}_i^{(m)} \equiv (\epsilon_i(\boldsymbol{\sigma}^{(m),1}), \ldots, \epsilon_i(\boldsymbol{\sigma}^{(m),a}), \ldots, \epsilon_i(\boldsymbol{\sigma}^{(m),n}))$$

The resulting replicated partition function reads [9]:

$$\overline{Z^n} \equiv \int dQ[\boldsymbol{J}^{\text{out}}] \sum_{\{\boldsymbol{\sigma}^{(1),a}\}} \sum_{\{\boldsymbol{\sigma}^{(2),a}\}} \prod_{\underline{\epsilon}} \delta_{Nc_1(\underline{\epsilon}), Nc_2(\underline{\epsilon})} \tag{4.10}$$
$$\exp\left\{ -\sum_{a=1}^n H(\boldsymbol{\sigma}^{(1),a}, \boldsymbol{\sigma}^{(2),a}; \boldsymbol{J}^{\text{out}}) + N \sum_{\underline{\epsilon}} c_1(\underline{\epsilon}) \log c_1(\underline{\epsilon}) \right\}$$

We briefly report here the main results of this approach for the gaussian channel described by Eq.(1.3). A detailed analysis will be presented elsewhere [9].

For recursive turbo codes there exists a low noise phase $w^2 < w_c^2$ where the error probability vanishes in the thermodynamic limit (i.e. for infinitely long sequences). In this phase the model is completely ordered:

$$\overline{\mathcal{P}}(\theta) = \delta(\theta - 1) \tag{4.11}$$

15

A local stability analysis yields the critical value w_{loc}^2 such that for $w^2 > w_{loc}^2$ the no-error phase is destroyed by small fluctuations. Clearly $w_{loc}^2 \geq w_c^2$. We computed w_{loc}^2 for the two cases listed below.

For both the rate is $R = 1/3$ so that the Shannon noise threshold as given by Eq.(1.4) is $w_{Shannon}^2 = 1/(2^{2/3} - 1) \simeq -2.31065$ db. Error free communication can take place only for $w^2 < w_{Shannon}^2$.

- For model (a), defined by Eqs.(2.12),(2.13) one gets $w_{loc}^2 = 1/\ln 4 \simeq 1.41855$ db.

- For model (b), defined by Eqs.(2.14),(2.15) one obtains $w_{loc}^2 = -1/(2 \ln x_c)$ where $x_c \simeq 0.741912\ldots$ is the only real solution of the equation

$$2x^5 + x^2 = 1 \tag{4.12}$$

The resulting value $w_{loc}^2 \simeq -2.23990$ db is quite near to the Shannon threshold.

5 Discussion.

We formulated turbo codes as a spin model Hamiltonian and we obtained new results using the replica method. It is well known that this method is not mathematically rigorous. So it is natural to question the validity of our results. For this purpose we have carried out numerical simulations of the following codes: the recursive turbo code corresponding to the convolutional code (a) of Sec.(2), its error probability is reported in Fig.(4); the non recursive turbo code obtained by permuting the generating polynomials of the previous one (see Fig.(5)); the recursive turbo code corresponding to the code (b) of the same section (see Fig.(6)). We used the Berrou et al. turbo decoding algorithm and averaged over 200 to 500 realizations of the disorder.

The first conclusion is that recursive turbo codes are much better codes than non recursive ones. Furthemore our results for recursive turbo codes are compatible with the existence of a threshold w_c^2 such that for $w^2 < w_c^2$ the error probability per bit is zero, while no such threshold seems to exist for non recursive codes. This is in agreement with replica theory. Zero error probability can only be achieved in the $N \to \infty$ limit. Our simulations are for $N = 10^5$. It would be interesting to perform a detailed study of finite size corrections, i.e. of the N dependence of the error probability per bit.

We now discuss the numerical value of the noise threshold w_c^2. The first remark is that both numerically and analytically, the critical value is below Shannon's bound and that it depends on the convolutional code (i.e. on the generating polynomials). The second remark is that the analytical value of thresold, $w_{loc}^2 \simeq 1.4186$ db is in very good agreement with the numerical value for the code (a). For code (b) $w_{loc}^2 \simeq -2.240$ db while one gets $w_c^2 \simeq -1.7$ db from the simulations. It would be interesting to understand this disagreement. As we said in the previous section, w_{loc} was calculated by a local stability analysis of the ordered phase, i.e. we assumed that the transition is of second order. A possible explanation would be that the transition is second order for code (a) and first order for code (b). Numerical results seem to support this hypothesis, as the variation of the error probability as a function of noise is much sharper in case (b). But a much more

16

careful analysis of finite size effects is necessary in order to settle this question numerically. One should also look analytically for the occurence of a first order transition.

Another important issue is the breaking of replica symmetry. Since turbo-decoding is thought to be an approximate algorithm, it may be not the best tool to look for replica symmetry breaking. We have started an analytical investigation of replica symmetry breaking.

References

[1] C.Berrou, A.Glavieux, and P.Thitimajshima. Proc.1993 Int.Conf.Comm. 1064-1070

[2] A.J.Viterbi. IEEE Trans.Com.Technology **COM-19**(1971) 751-771

[3] L.Bahl, J.Cocke, F.Jelinek, and J.Raviv. IEEE Trans.Inf.Theory **IT-20**(1974) 248-287

[4] N.Sourlas. Nature **339**(1989) 693-694
N.Sourlas, in *Statistical Mechanics of Neural Networks*, Lecture Notes in Physics 368, ed. L. Garrido, Springer Verlag (1990)
N.Sourlas, Ecole Normale Supérieure preprint (April 1993)
N.Sourlas, in *From Statistical Physics to Statistical Inference and Back,* ed. P. Grassberger and J.-P. Nadal, Kluwer Academic (1994), page 195.

[5] B.Derrida. Phys.Rev. **B 24**(1981) 2613-2626

[6] P.Ruján. Phys.Rev.Lett. **70**(1993) 2968-2971
N.Sourlas. Europhys.Lett. **25**(1994) 159-164
H.Nishimori. J. Phys. Soc. Jpn. **62**(1993) 2973

[7] H.Nishimori. J.Phys. **C 13**(1980) 4071-4076

[8] R.Monasson. J.Phys. **A31**(1998) 513-529

[9] A.Montanari, in preparation.
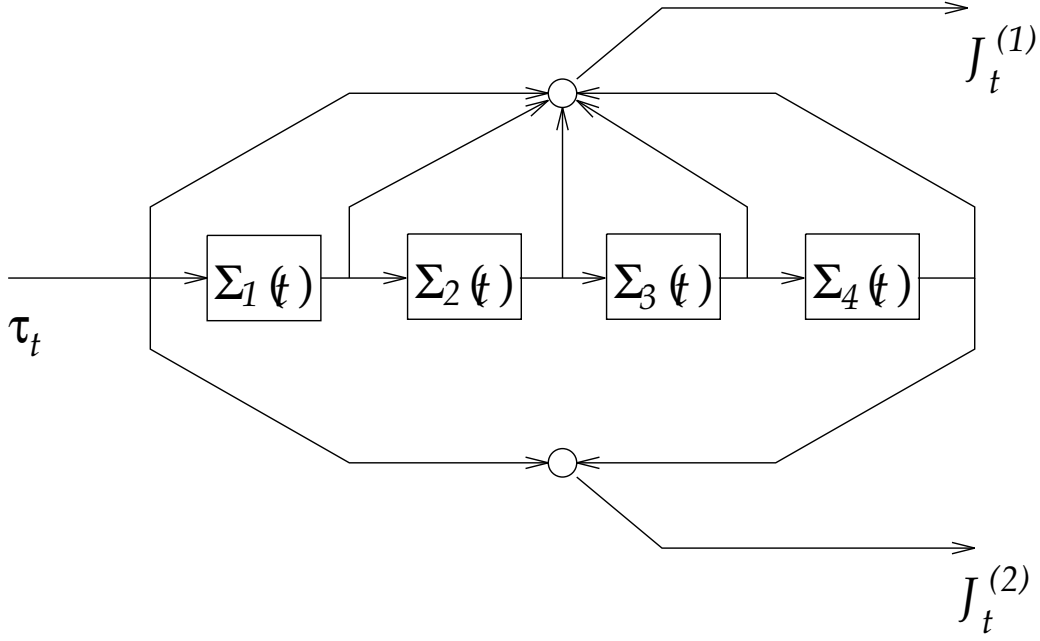
Figure 1: Schematic representation of the encoder for a non recursive convolutional code called code (c) in the text and defined by Eqs.(2.16),(2.17).
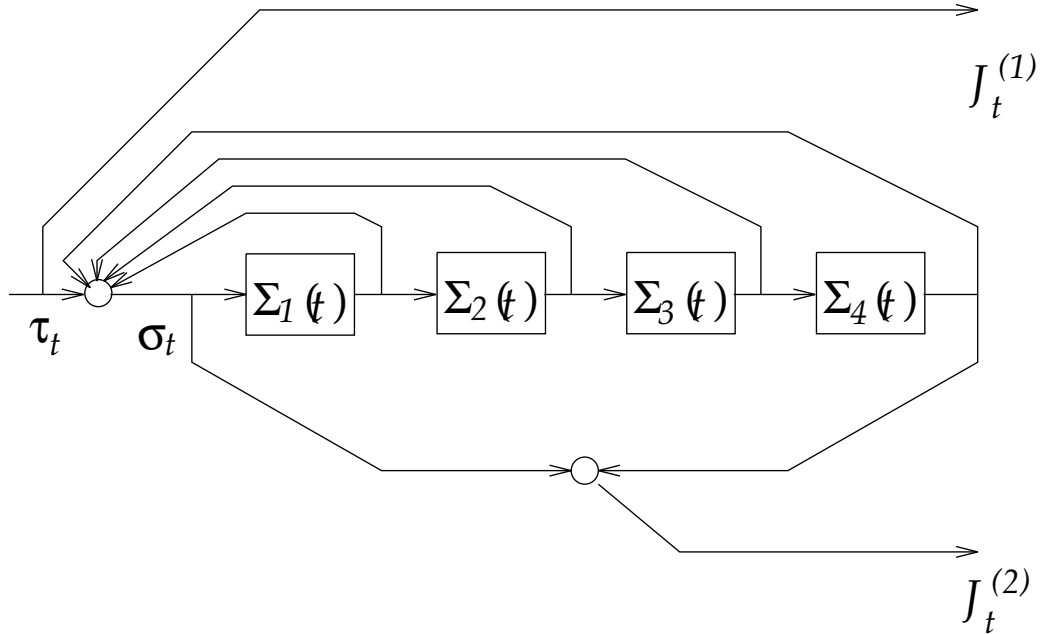

Figure 2: The encoder for the recursive convolutional code with the same generating polynomials as in Fig.(1) (cfr. Eqs.(2.16),(2.17)).
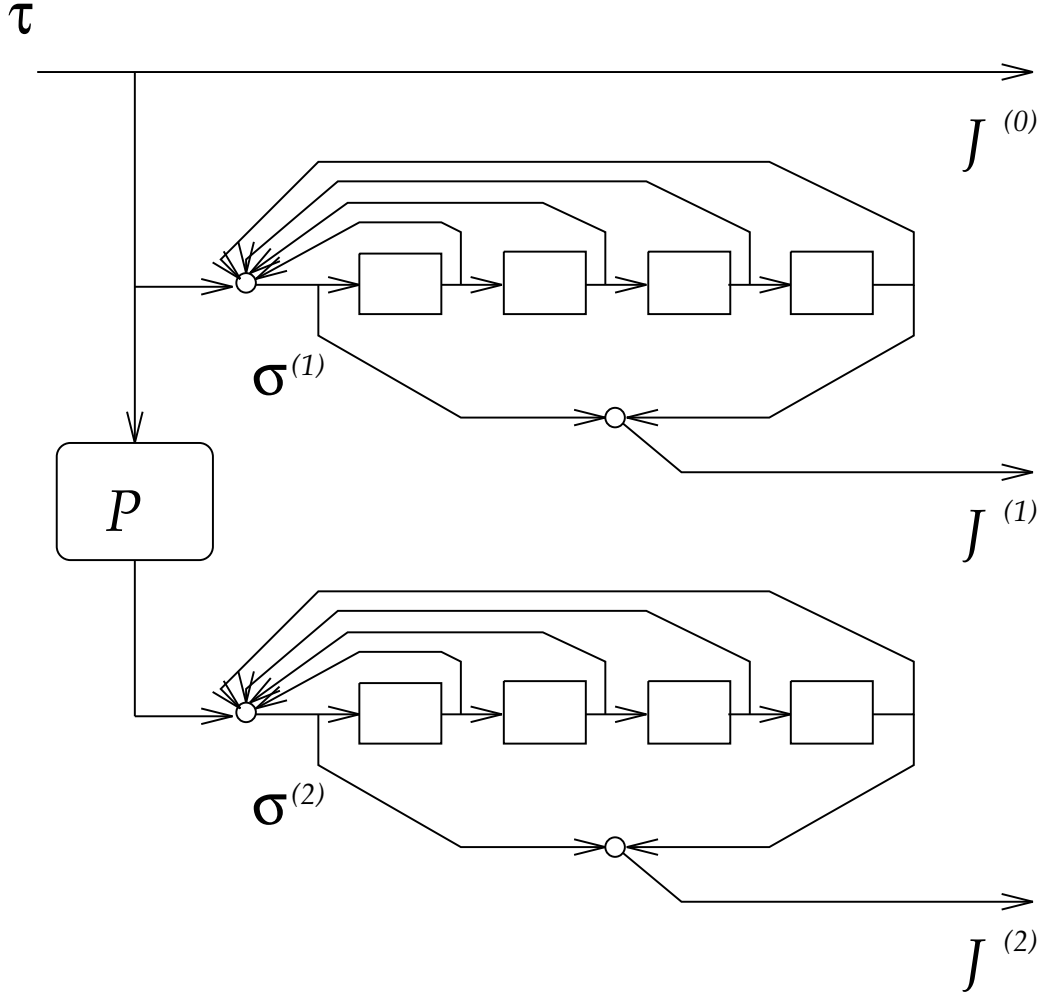
Figure 3: Schematic representation of the encoder for a recursive turbo code with generating polynomials as in the previous figures (cfr. Eqs.(2.16),(2.17)). Notice the presence of the interleaver (denoted by P) which implements the permutation.
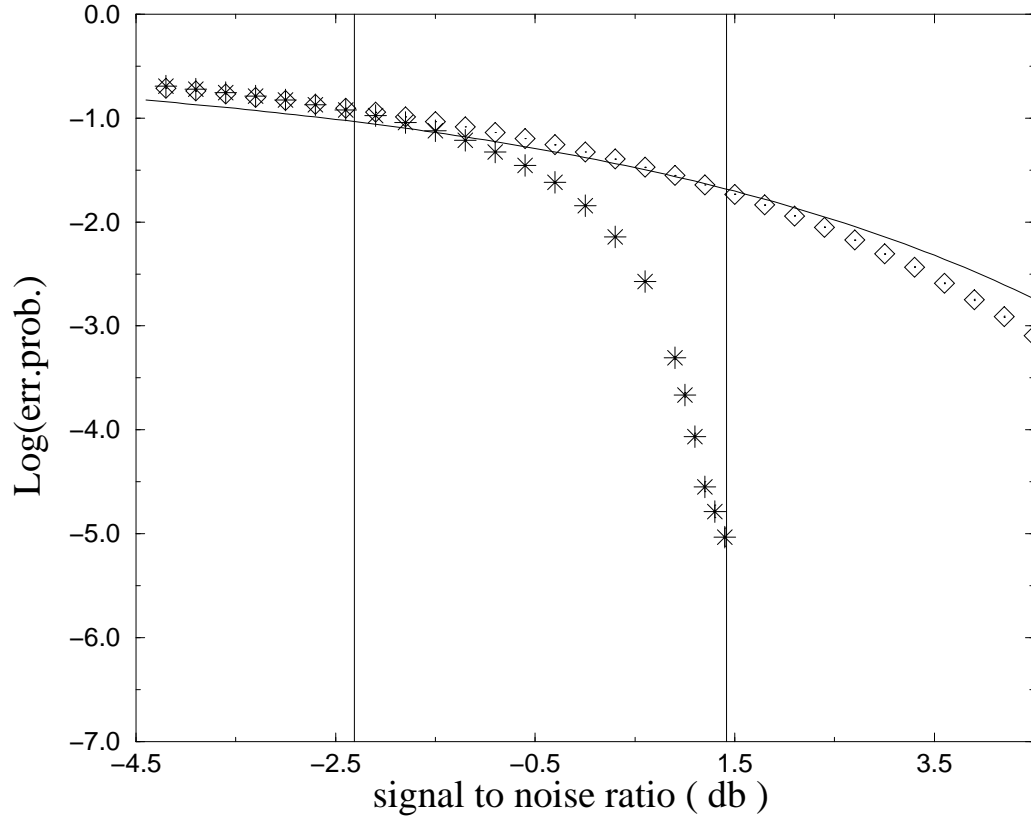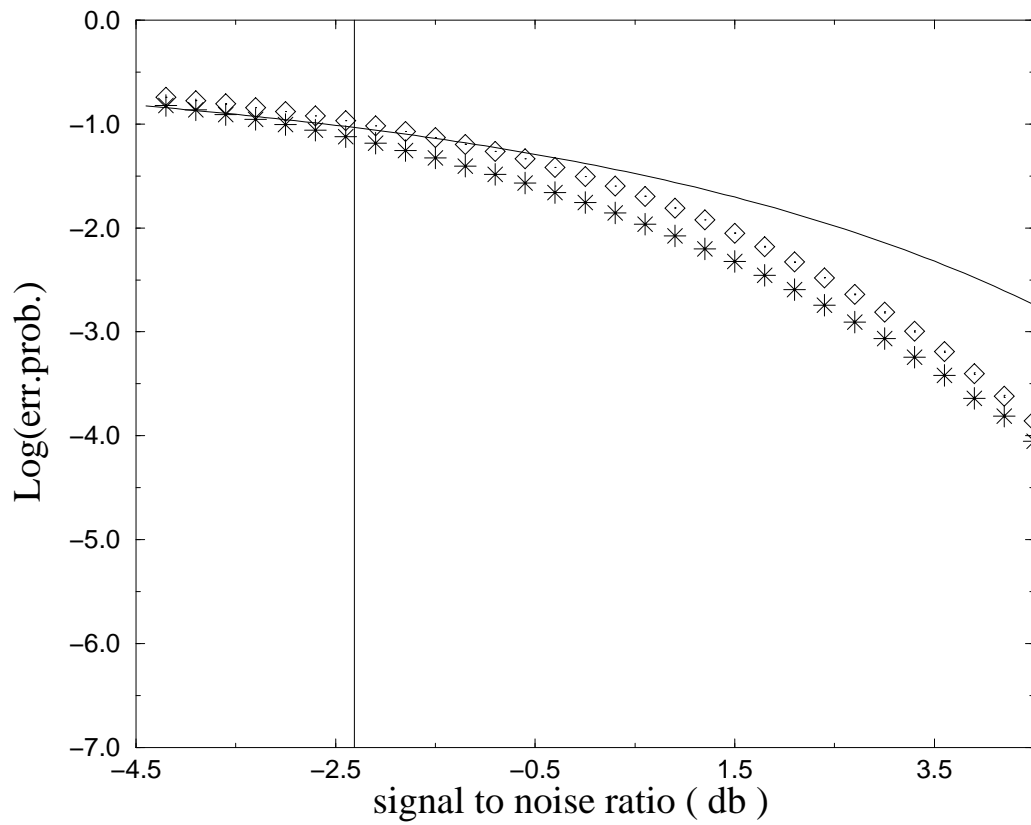
Figure 4: Numerical results for the error probability per bit of the recursive turbo code buildt from the convolutional code (a) (cfr. Eqs.(2.12),(2.13)). Stars ($*$) refer to the turbo code, diamonds ($\diamond$) to the convolutional code obtained by setting the permutation P equal to the identity permutation, and the continuous line to the uncoded message. The leftmost vertical line is located at the Shannon thresold ($w^2 = w^2_{Shannon}$) while the rightmost at the thresold of local stability ($w^2 = w^2_{loc}$, see Sec.(4)).

Figure 5: Numerical results for the error probability per bit of the non recursive turbo code buildt from the convolutional code (a) (cfr. Eqs.(2.12),(2.13)). The symbols have the same meaning explained in the caption of Fig.(4).
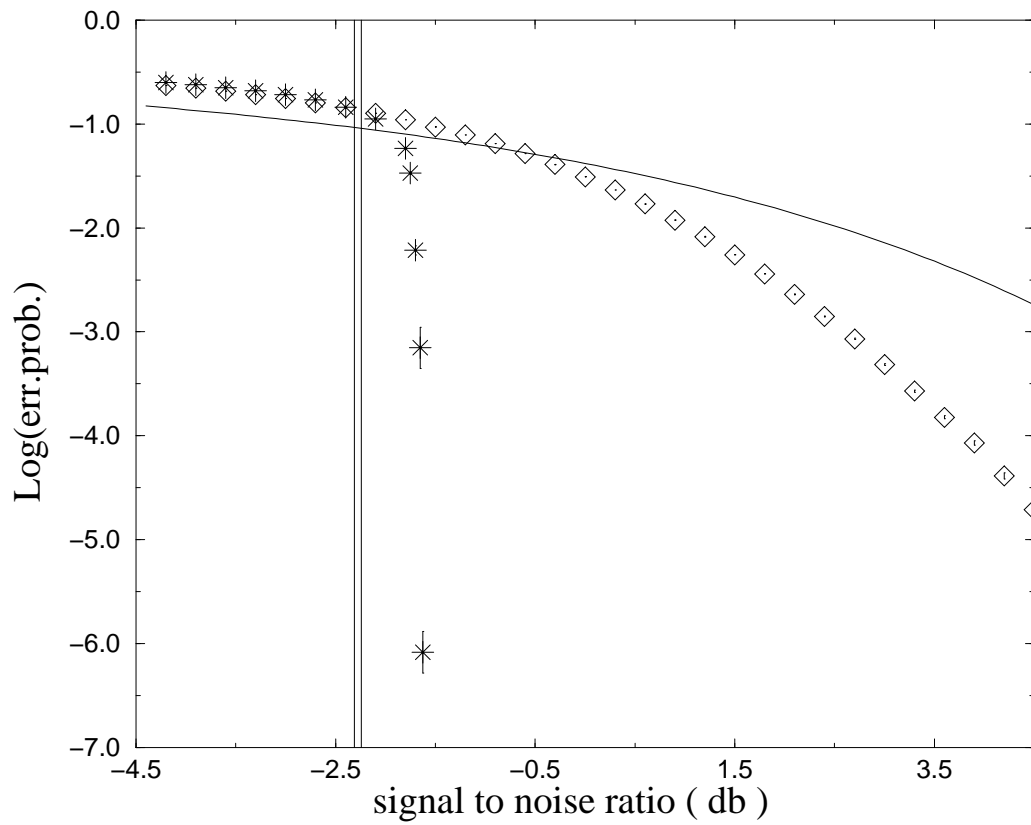
Figure 6: Numerical results for the error probability per bit of the recursive turbo code buildt from the convolutional code (b) (cfr. Eqs.(2.14),(2.15)). For the meaning of various symbols see the caption of Fig.(4). Notice that in this case $w_{loc}^2 \sim w_{Shannon}^2$.