# Modern Coding Theory: The Statistical Mechanics and Computer Science Point of View

Andrea Montanari[1] and Rüdiger Urbanke[2]

[1]Stanford University, montanari@stanford.edu, [2]EPFL, ruediger.urbanke@epfl.ch *

February 12, 2007

**Abstract**

These are the notes for a set of lectures delivered by the two authors at the Les Houches Summer School on 'Complex Systems' in July 2006. They provide an introduction to the basic concepts in modern (probabilistic) coding theory, highlighting connections with statistical mechanics. We also stress common concepts with other disciplines dealing with similar problems that can be generically referred to as 'large graphical models'.

While most of the lectures are devoted to the classical channel coding problem over simple memoryless channels, we present a discussion of more complex channel models. We conclude with an overview of the main open challenges in the field.

## 1 Introduction and Outline

The last few years have witnessed an impressive convergence of interests between disciplines which are *a priori* well separated: coding and information theory, statistical inference, statistical mechanics (in particular, mean field disordered systems), as well as theoretical computer science. The underlying reason for this convergence is the importance of probabilistic models and/or probabilistic techniques in each of these domains. This has long been obvious in information theory [53], statistical mechanics [10], and statistical inference [45]. In the last few years it has also become apparent in coding theory and theoretical computer science. In the first case, the invention of Turbo codes [7] and the re-invention of Low-Density Parity-Check (LDPC) codes [30, 28] has motivated the use of random constructions for coding information in robust/compact ways [50]. In the second case (theoretical computer science) the relevance of randomized algorithms has steadily increased (see for instance [41]), thus motivating deep theoretical developments. A particularly important example is provided by the Monte Carlo Markov Chain method for counting and sampling random structures.

Given this common probabilistic background, some analogies between these disciplines is not very surprising nor is it particularly interesting. The key new ingredient which lifts the connections beyond some superficial commonalities is that one can name specific problems, questions, and results which lie at the intersection of these fields while being of central interest for each of them. The set of problems and techniques thus defined can be somewhat loosely named "theory of large graphical models." The typical setting is the following: a large set of random variables taking values in a finite (typically quite small) alphabet with a "local" dependency structure; this local dependency structure is conveniently described by an appropriate graph.

*The work of A. Montanari was partially supported by the European Union under the project EVERGROW. The work of R. Urbanke was partially supported by the NCCR-MICS, a center supported by the Swiss National Science Foundation under grant number 5005-67322.

In this lecture we shall use "modern" coding theory as an entry point to the domain. There are several motivations for this: (*i*) theoretical work on this topic is strongly motivated by concrete and well-defined practical applications; (*ii*) the probabilistic approach mentioned above has been quite successful and has substantially changed the field (whence the reference to *modern* coding theory); (*iii*) a sufficiently detailed picture exists illustrating the interplay among different view points.

We start in Section 2 with a brief outline of the (channel coding) problem. This allows us to introduce the standard definitions and terminology used in this field. In Section 3 we introduce ensembles of codes defined by sparse random graphs and discuss their most basic property – the weight distribution. In Section 4 we phrase the decoding problem as an inference problem on a graph and consider the performance of the efficient (albeit in general suboptimal) message-passing decoder. We show how the performance of such a combination (sparse graph code and message-passing decoding) can be analyzed and we discuss the relationship of the performance under message-passing decoding to the performance of the optimal decoder. In Section 5 we briefly touch on some problems beyond coding, in order to show as similar concept emerge there. In particular, we discuss how message passing techniques can be successfully used in some families of counting/inference problems. In Section 6 we show that several of the simplifying assumptions (binary case, symmetry of channel, memoryless channels) are convenient in that they allow for a simple theory but are not really necessary. In particular, we discuss a simple channel with memory and we see how to proceed in the asymmetric case. Finally, we conclude in Section 7 with a few fundamental open problems.

To readers who would like to find current contributions on this topic we recommend the *IEEE Transactions on Information Theory*. A considerably more in-depth discussion can be found in the two upcoming books *Information, Physics and Computation* [36] and *Modern Coding Theory* [50]. Standard references on coding theory are [6, 9, 26] and very readable introductions to information theory can be found in [12, 20]. Other useful reference sources are the book by Nishimori [44] as well as the book by MacKay [29].

# 2 Background: The Channel Coding Problem

The central problem of communications is how to transmit information reliably through a noisy (and thus unreliable) communication channel. Coding theory aims at accomplishing this task by adding a properly designed redundancy to the transmitted message. This redundancy is then used at the receiver to reconstruct the original message despite the noise introduced by the channel.

## 2.1 The Problem

In order to model the situation described above we shall assume that the noise is random with some known distribution.[1] To keep things simple we shall assume that the communication channel admits as input binary symbols $x \in \{0, 1\}$, while the output belongs to some finite alphabet $\mathcal{A}$. We denote the probability of observing the output $y \in \mathcal{A}$ given that the input was $x \in \{0, 1\}$ by $Q(y|x)$. The channel model is defined by the transition probability matrix

$$Q = \{Q(y|x) : x \in \{0, 1\},\ y \in \mathcal{A}\}\,. \tag{2.1}$$

Of course, the entries of this matrix must be non-negative and normalized in such a way that $\sum_y Q(y|x) = 1$. It is convenient to have a few simple examples in mind. We refer to Fig. 1 for an illustration of the channel models which we introduce in the following three examples.

Example 1: The *binary symmetric channel* BSC($p$) is defined by letting $\mathcal{A} = \{0, 1\}$ and $Q(0|0) = Q(1|1) = 1-p$; the normalization then enforces $Q(1|0) = Q(0|1) = p$. In words, the channel "flips" the input bit with probability $p \in [0, 1]$. Since flips are introduced for each bit independently we

[1] It is worth mentioning that an alternative approach would be to consider the noise as 'adversarial' (or worst case) under some constraint on its intensity.

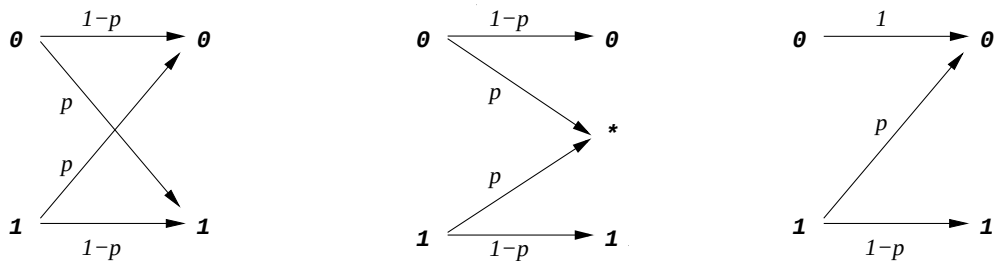


Figure 1: Schematic description of three simple binary memoryless channels. From left to right: binary symmetric channel BSC($p$), binary erasure channel BEC($\epsilon$), and Z channel ZC($p$).

say that the channel is *memoryless*. Except for an example in Section 6.2 all channels which we consider are memoryless.

Example 2: The *binary erasure channel* BEC($\epsilon$) is defined by $\mathcal{A} = \{0, 1, *\}$ and $Q(0|0) = Q(1|1) = 1 - \epsilon$ while $Q(*|0) = Q(*|1) = \epsilon$. In words, the channel input is erased with probability $\epsilon$ and it is transmitted correctly otherwise.

Example 3: The *Z-channel* ZC($p$) has an output alphabet $\mathcal{A} = \{0, 1\}$ but acts differently on input 0 (that is transmitted correctly) and 1 (that is flipped with probability $p$). We invite the reader to write the transition probability matrix.

Since in each case the input is binary we speak of a *binary-input* channel. Since further in all models each input symbol is distorted independently from all other ones we say that the channels are *memoryless*. It is convenient to further restrict our attention to *symmetric* channels: this means that there is an involution on $\mathcal{A}$ (i.e. a mapping $\iota : \mathcal{A} \to \mathcal{A}$ such that $\iota \circ \iota = 1$) so that $Q(y|0) = Q(\iota(y)|1)$. (E.g., if $\mathcal{A} = \mathbb{R}$ then we could require that $Q(y|0) = Q(-y|1)$.) This condition is satisfied by the first two examples above but not by the third one. To summarize these three properties one refers to such models as BMS channels.

In order to complete the problem description we need to formalize the information which is to be transmitted. We shall model this probabilistically as well and assume that the transmitter has an information source that provides an infinite stream of i.i.d. fair coins: $\{z_i; i = 0, 1, 2, \dots\}$, with $z_i \in \{0, 1\}$ uniformly at random. The goal is to reproduce this stream faithfully after communicating it over the noisy channel.

Let us stress that, despite its simplification, the present setting contains most of the crucial and challenges of the channel coding problem. Some of the many generalizations are described in Section 6.

## 2.2 Block Coding

The (general) coding strategy we shall consider here is *block coding*. It works as follows:

- The source stream $\{z_i\}$ is chopped into blocks of length $L$. Denote one such block by $\underline{z}$, $\underline{z} = (z_1, \dots, z_L) \in \{0, 1\}^L$.
- Each block is fed into an *encoder*. This is a map $\mathsf{F} : \{0, 1\}^L \to \{0, 1\}^N$, for some fixed $N > L$ (the *blocklength*). In words, the encoder introduces redundancy in the source message. Without loss of generality we can assume $\mathsf{F}$ to be injective. It this was not the case, even in the absence of noise, we could not uniquely recover the transmitted information from the observed codeword.
- The image of $\{0, 1\}^L$ under the map $\mathsf{F}$ is called the *codebook*, or sometimes the *code*, and it will be denoted by $\mathfrak{C}$. The code contains $|\mathfrak{C}| = 2^L$ strings of length $N$ called *codewords*. These are the possible channel inputs. The codeword $\underline{x} = \mathsf{F}(\underline{z})$ is sent through the channel, bit by bit.


Figure 2: Flow chart of a block coding scheme.

- Let $\underline{y} = (y_1, \cdots, y_N) \in \mathcal{A}^N$ be the channel output. Conditioned on $\underline{x}$ the $y_i$, $i = 1, \cdots, L$, are independent random variables with distribution $y_i \stackrel{\mathrm{d}}{=} Q(\cdot\,|x_i)$ (here and below $\stackrel{\mathrm{d}}{=}$ denotes identity in distribution and $x \stackrel{\mathrm{d}}{=} P(\,\cdot\,)$ means that $x$ is a random variable with distribution $P(\,\cdot\,)$).
- The channel output is fed into a *decoder*, which is a map $\widehat{\mathsf{F}} : \mathcal{A}^N \to \{0,1\}^L$. It is the objective of the decoder to reconstruct the source $\underline{z}$ from the noisy channel output $\underline{y}$.

The flow chart describing this coding scheme is shown in Fig. 2. It is convenient to slightly modify the above scheme. Notice that, under the hypothesis that the encoder is injective, the codebook is in one-to-one correspondence with the source sequences. Since these are equiprobable, the transmitted codewords are equiprobable as well. We can therefore equivalently assume that the transmitter picks a codeword uniformly at random and transmits it. Every reference to the source stream can be eliminated if we redefine the decoder to be a map $\widehat{\mathsf{F}} : \mathcal{A}^N \to \{0,1\}^N$, i.e., the decoder aims to reconstruct the transmitted codeword. If $\widehat{\mathsf{F}}(\underline{y}) \notin \mathfrak{C}$ we declare an error.[2] In the following we shall also use the notation $\widehat{\mathsf{F}}(\underline{y}) = \underline{\widehat{x}}(\underline{y}) = (\widehat{x}_1(\underline{y}), \dots, \widehat{x}_N(\underline{y}))$.

One crucial parameter of a code is its *rate*: it quantifies how many bits of information are transmitted per channel use,

$$R \equiv \frac{L}{N} = \frac{1}{N} \log_2 |\mathfrak{C}| \,. \tag{2.2}$$

Two fundamental performance parameters are the *bit* (or 'symbol') and *block* (or 'word') *error rates*. The block error rate is the probability that the input codeword is not recovered correctly at the end of the process,

$$\mathrm{P_B} \equiv \mathbb{P}\left\{\underline{\widehat{x}}(\underline{y}) \neq \underline{x}\right\} \,. \tag{2.3}$$

The bit error rate is the expected fraction of bits that are not recovered correctly,

$$\mathrm{P_b} \equiv \frac{1}{N} \sum_{i=1}^{N} \mathbb{P}\left\{\widehat{x}_i(\underline{y}) \neq x_i\right\} \,. \tag{2.4}$$

It should not be too surprising that one can trade-off rate and error probability. We want to achieve a high rate and achieve a low probability of error. However, increasing the rate decreases the redundancy built into the codeword, thus inducing a higher error probability. The aim of coding theory is to choose the code $\mathfrak{C}$ and the decoding function $\underline{\widehat{x}}(\,\cdot\,)$ in a way to optimize this trade-off.

[2] More precisely, if we are interested only in the block probability of error, i.e., the frequency at which the whole block of data is decoded correctly, then indeed any one-to-one mapping between information word and codeword performs identical. If, on the other hand, we are interested in the fraction of *bits* that we decode correctly then the exact mapping from information word to codeword does come into play. We shall ignore this somewhat subtle point in the sequel.

## 2.3 Decoding

Given the code there is a simple (although in general not computationally efficient) prescription for the decoder. If we want to minimize the block error rate, we must chose the most likely codeword,

$$\widehat{\underline{x}}^{\rm B}(\underline{y}) \equiv \arg\max_{\underline{x}} \mathbb{P}\{\underline{X}=\underline{x}|\underline{Y}=\underline{y}\}\,. \tag{2.5}$$

To minimize the bit error rate we must instead return the sequence of most likely bits,

$$\widehat{x}_i^{\rm b}(\underline{y}) \equiv \arg\max_{x_i} \mathbb{P}\{X_i=x_i|\underline{Y}=\underline{y}\}\,. \tag{2.6}$$

The reason of these prescriptions is the object of the next exercise.

Exercise 1: Let $(U,V)$ be a pair of discrete random variables. Think of $U$ as a 'hidden' variable and imagine you observe $V=v$. We want to understand what is the optimal estimate for $U$ given $V=v$. Show that the function $v\mapsto\widehat{u}(v)$ that minimizes the error probability $\mathsf{P}(\widehat{u})\equiv\mathbb{P}\{U\neq\widehat{u}(V)\}$ is given by

$$\widehat{u}(v) = \arg\max_{u} \mathbb{P}\{U=u|V=v\}\,. \tag{2.7}$$

It is instructive to explicitly write down the conditional distribution of the channel input given the output. We shall denote it as $\mu_{\mathfrak{C},y}(\underline{x})=\mathbb{P}\{\underline{X}=\underline{x}|\underline{Y}=\underline{y}\}$ (and sometimes drop the subscripts $\mathfrak{C}$ and $y$ if they are clear from the context). Using Bayes rule we get

$$\mu_{\mathfrak{C},y}(\underline{x}) = \frac{1}{Z(\mathfrak{C},y)}\prod_{i=1}^{N} Q(y_i|x_i)\,\mathbb{I}_{\mathfrak{C}}(\underline{x})\,, \tag{2.8}$$

where $\mathbb{I}_{\mathfrak{C}}(\underline{x})$ denotes the code membership function ($\mathbb{I}_{\mathfrak{C}}(\underline{x})=1$ if $\underline{x}\in\mathfrak{C}$ and $=0$ otherwise).

According to the above discussion, decoding amounts to computing the marginals (for symbol MAP) or the mode[3] (for word MAP) of $\mu(\,\cdot\,)$. More generally, we would like to understand the properties of $\mu(\,\cdot\,)$: is it concentrated on a single codeword or spread over many of them? In the latter case, are these close to each other or very different? And what is their relationship with the transmitted codeword?

The connection to statistical mechanics emerges in the study of the decoding problem [56, 51]. To make it completely transparent we rewrite the distribution $\mu(\,\cdot\,)$ in Boltzmann form

$$\mu_{\mathfrak{C},y}(\underline{x}) = \frac{1}{Z(\mathfrak{C},y)}\,e^{-E_{\mathfrak{C},y}(\underline{x})}\,, \tag{2.9}$$

$$E_{\mathfrak{C},y}(\underline{x}) = \begin{cases} -\sum_{i=1}^{N}\log Q(y_i|x_i), & \text{if } \underline{x}\in\mathfrak{C}\,,\\ +\infty, & \text{otherwise}\,. \end{cases} \tag{2.10}$$

The word MAP and bit MAP rule can then be written as

$$\widehat{\underline{x}}^{\rm B}(\underline{y}) = \arg\min_{\underline{x}} E_{\mathfrak{C},y}(\underline{x})\,, \tag{2.11}$$

$$\widehat{x}_i^{\rm b}(\underline{y}) = \arg\max_{x_i}\sum_{x_j:j\neq i}\mu_{\mathfrak{C},y}(\underline{x})\,. \tag{2.12}$$

In words, word MAP amounts to computing the ground state of a certain energy function, and bit MAP corresponds to computing the expectation with respect to the Boltzmann distribution. Notice furthermore that $\mu(\,\cdot\,)$ is itself random because of the randomness in $y$ (and we shall introduce further randomness in the choice of the code). This is analogous to what happens in statistical physics of disordered systems, with $\underline{y}$ playing the role of quenched random variables.

[3] We recall that the mode of a distribution with density $\mu(\,\cdot\,)$ is the value of $x$ that maximizes $\mu(x)$.

## 2.4 Conditional Entropy and Free Energy

As mentioned above, we are interested in understanding the properties of the (random) distribution $\mu_{\mathfrak{C},y}(\,\cdot\,)$. One possible way of formalizing this idea is to consider the entropy of this distribution.

Let us recall that the (Shannon) entropy of a discrete random variable $X$ (or, equivalently, of its distribution) quantifies, in a very precise sense, the 'uncertainty' associated with $X$.[4] It is given by

$$H(X) = -\sum_x \mathbb{P}(x)\log\mathbb{P}(x)\,. \tag{2.13}$$

For two random variables $X$ and $Y$ one defines the conditional entropy of $X$ given $Y$ as

$$H(X|Y) \;=\; -\sum_{x,y}\mathbb{P}(x,y)\log\mathbb{P}(x|y) = \mathbb{E}_y\left\{-\sum_x \mathbb{P}(x|Y)\log\mathbb{P}(x|Y)\right\}. \tag{2.14}$$

This quantifies the remaining uncertainty about $X$ when $Y$ is observed.

Considering now the coding problem. Denote by $\underline{X}$ the (uniformly random) transmitted codeword and by $\underline{Y}$ the channel output. The right-most expression in Eq. (2.14) states that $H(\underline{X}|\underline{Y})$ is the expectation of the entropy of the conditional distribution $\mu_{\mathfrak{C},y}(\,\cdot\,)$ with respect to $\underline{y}$.

Let us denote by $\nu_{\mathfrak{C}}(x)$ the probability that a uniformly random codeword in $\mathfrak{C}$ takes the value $x$ at the $i$-th position, averaged over $i$. Then a straightforward calculation yields

$$H(\underline{X}|\underline{Y}) \;=\; -\frac{1}{|\mathfrak{C}|}\sum_{\underline{x},\underline{y}}\prod_{i=1}^N Q(y_i|x_i)\,\log\left\{\frac{1}{Z(\mathfrak{C},\underline{y})}\prod_{i=1}^N Q(y_i|x_i)\right\}, \tag{2.15}$$

$$\;=\; -N\sum_{x,y}\nu_{\mathfrak{C}}(x)Q(y|x)\log Q(y|x) + \mathbb{E}_{\underline{y}}\log Z(\mathfrak{C},\underline{y})\,. \tag{2.16}$$

The 'type' $\nu_{\mathfrak{C}}(x)$ is usually a fairly straightforward characteristic of the code. For most of the examples considered below we can take $\nu_{\mathfrak{C}}(\mathtt{0}) = \nu_{\mathfrak{C}}(\mathtt{1}) = 1/2$. As a consequence the first of the terms above is trivial to compute (it requires summing over $2|\mathcal{A}|$ terms).

On the other hand the second term is highly non-trivial. The reader will recognize the expectation of a free energy, with $\underline{y}$ playing the role of a quenched random variable.

The conditional entropy $H(\underline{X}|\underline{Y})$ provides an answer to the question: how many codewords is $\mu_{\mathfrak{C},y}(\,\cdot\,)$ spread over? It turns out that about $e^{H(\underline{X}|\underline{Y})}$ of them carry most of the weight.

## 2.5 Shannon Theorem and Random Coding

As mentioned above, there exists an obvious tradeoff between high rate and low error probability. In his celebrated 1948 paper [53], Shannon derived the optimal error probability-vs-rate curve in the limit of large blocklengths. In particular, he proved that if the rate is larger than a particular threshold, then the error probability can be made arbitrarily small. The threshold depends on the channel and it is called the channel *capacity*. The capacity of a BMS channel (measured in bits per channel use) is given by the following elementary expression,

$$\begin{aligned}\mathtt{C}(Q) =& H(X) - H(X|Y)\\ =& 1 + \sum_y Q(y|\mathtt{0})\log_2\left\{\frac{Q(y|\mathtt{0})}{Q(y|\mathtt{0})+Q(y|\mathtt{1})}\right\}.\end{aligned}$$

For instance, the capacity of a BSC($p$) is $\mathtt{C}(p) = 1-\mathfrak{h}_2(p)$, (where $\mathfrak{h}_2(p) = -p\log_2 p-(1-p)\log_2(1-p)$ is the entropy of a Bernoulli random variable of parameter $p$) while the capacity of a BEC($\epsilon$) is $\mathtt{C}(\epsilon) = 1-\epsilon$. As an illustration, the capacity of a BSC($p$) with flip probability $p \approx 0.110028$ is $\mathtt{C}(p) = 1/2$: such a channel can be used to transmit reliably $1/2$ bit of information per channel use.

[4] For a very readable account of information theory we recommend [12].

**Theorem 2.1** (Channel Coding Theorem)**.** *For any BMS channel with transition probability $Q$ and $R < \mathtt{C}(Q)$ there exists a sequence of codes $\mathfrak{C}_N$ of increasing blocklength $N$ and rate $R_N \to R$ whose block error probability $\mathrm{P}_{\mathrm{B}}^{(N)} \to 0$ as $N \to \infty$.*

*Vice versa, for any $R > \mathtt{C}(Q)$ the block error probability of a code with rate at least $R$ is bounded away from 0.*

The prove of the first part ('achievability') is one of the first examples of the so-called 'probabilistic method'. In order to prove that there exists an object with a certain property (a code with small error probability), one constructs a probability distribution over all potential candidates (all codes of a certain blocklength and rate) and shows that a random element has the desired property with non-vanishing probability. The power of this approach is in the (meta-mathematical) observation that random constructions are often much easier to produce than explicit, deterministic ones.

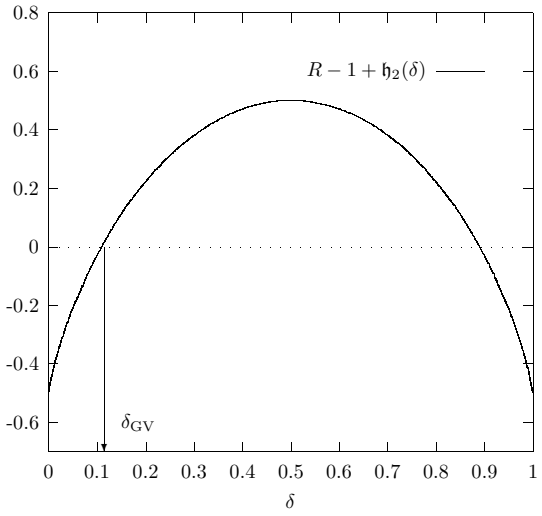


Figure 3: Exponential growth rate for the expected distance enumerator $\mathbb{E}\mathcal{N}_{\underline{x}^{(0)}}(n\delta)$ within the random code ensemble.

The distribution over codes proposed by Shannon is usually referred to as the *random code* (or, *Shannon*) *ensemble*, and is particularly simple. One picks a code $\mathfrak{C}$ uniformly at random among all codes of blocklength $N$ and rate $R$. More explicitly, one picks $2^{NR}$ codewords as uniformly random points in the hypercube $\{\mathtt{0},\mathtt{1}\}^N$. This means that each codeword is a string of $N$ fair coins $\underline{x}^{(\alpha)} = (x_1^{(\alpha)}, \dots, x_N^{(\alpha)})$ for[5] $\alpha = 1, \dots, 2^{NR}$.

Once the ensemble is defined, one can estimate its average block error probability and show that it vanishes in the blocklength for $R < \mathtt{C}(Q)$. Here we will limit ourselves to providing some basic 'geometric' intuition of why a random code from the Shannon ensemble performs well with high probability.[6]

Let us consider a particular codeword, say $\underline{x}^{(0)}$, and try to estimate the distance (from $\underline{x}^{(0)}$) at which

[5]The reader might notice two imprecisions with this definition. First, $2^{NR}$ is not necessarily an integer: one should rather use $\lceil 2^{NR} \rceil$ codewords, but the difference is obviously negligible. Second, in contradiction with our definition, two codewords may coincide if they are independent. Again, only an exponentially small fraction of codewords will coincide and they can be neglected for all practical purposes.

[6]Here and in the rest of the lectures, the expression *with high probability* means 'with probability approaching one as $N \to \infty$'

other codewords in $\mathfrak{C}$ can be found. This information is conveyed by the *distance enumerator*

$$\mathcal{N}_{\underline{x}^{(0)}}(d) \equiv \# \left\{ \underline{x} \in \mathfrak{C} \backslash \underline{x}^{(0)} \ \text{ such that } \ d(\underline{x}, \underline{x}^{(0)}) = d \right\} , \tag{2.17}$$

where $d(\underline{x}, \underline{x}')$ is the Hamming distance between $\underline{x}$ and $\underline{x}'$ (i.e., the number of positions in which $\underline{x}$ and $\underline{x}'$ differ). The expectation of this quantity is the number of codewords different from $\underline{x}^{(0)}$ (that is $(2^{NR}-1)$) times the probability that any given codeword has distance $d$ from $\underline{x}^{(0)}$. Since each entry is independent and different with probability $1/2$, we get

$$\mathbb{E}\mathcal{N}_{\underline{x}^{(0)}}(d) = (2^{NR}-1)\ \frac{1}{2^N}\binom{N}{d} \doteq 2^{N[R-1+\mathfrak{h}_2(\delta)]} , \tag{2.18}$$

where $\delta = d/N$ and $\doteq$ denotes equality to the leading exponential order.[7]

The exponent $R-1+\mathfrak{h}_2(\delta)$ is plotted in Fig. 3. For $\delta$ sufficiently small (and $R<1$) this exponent is negative. Its first zero, to be denoted as $\delta_{\rm GV}(R)$, is called the Gilbert-Varshamov distance. For any $\delta < \delta_{\rm GV}(R)$ the expected number of codewords of distance at most $N\delta$ from $\underline{x}^{(0)}$ is exponentially small in $N$. It follows that the probability to find *any codeword* at distance smaller than $N\delta$ is exponentially small in $N$.

Vice-versa, for $d = N\delta$, with $\delta > \delta_{\rm GV}(R)$, $\mathbb{E}\mathcal{N}_{\underline{x}^{(0)}}(d)$ is exponentially large in $N$. Indeed, $\mathcal{N}_{\underline{x}^{(0)}}(d)$ is a binomial random variable, because each of the $2^{NR}-1$ codewords is at distance $d$ independently and with the same probability. As a consequence, $\mathcal{N}_{\underline{x}^{(0)}}(d)$ is exponentially large as well with high probability.

The bottom line of this discussion is that, for any given codeword $\underline{x}^{(0)}$ in $\mathfrak{C}$, the closest other codeword is, with high probability, at distance $N(\delta_{\rm GV}(R) \pm \varepsilon)$. A sketch of this situation is provided in Fig. 4.

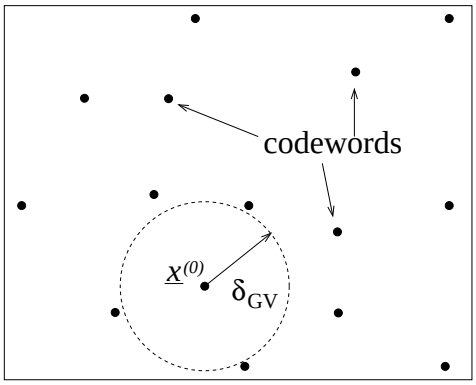


Figure 4: Pictorial description of a typical code from the random code ensemble.

Let us assume that the codeword $\underline{x}^{(0)}$ is transmitted through a BSC($p$). Denote by $\underline{y} \in \{0,1\}^N$ the channel output. By the law of large numbers $d(\underline{x}, \underline{y}) \approx Np$ with high probability. The receiver tries to reconstruct the transmitted codeword from $\underline{y}$ using word MAP decoding. Using Eq. (2.10), we see that the 'energy' of a codeword $\underline{x}^{(\alpha)}$ (or, in more conventional terms, its log-likelihood) is given by

$$E(\underline{x}^{(\alpha)}) = -\sum_{i=1}^{N} \log Q(y_i|x_i) = -\sum_{i=1}^{N} \Big\{ \mathbb{I}(y_i = x_i^{(\alpha)}) \log(1-p) + \mathbb{I}(y_i \neq x_i^{(\alpha)}) \log p \Big\} \tag{2.19}$$

$$= NA(p) + 2B(p) d(\underline{x}^{(\alpha)}, \underline{y}) , \tag{2.20}$$

[7] Explicitly, we write $f_N \doteq g_N$ if $\frac{1}{N}\log f_N/g_N \to 0$.

where $A(p) \equiv -\log p$ and $B(p) \equiv \frac{1}{2}\log(1-p)/p$. For $p < 1/2$, $B(p) > 0$ and word MAP decoding amounts to finding the codeword $\underline{x}^{(\alpha)}$ which is closest in Hamming distance to the channel output $\underline{y}$. By the triangle inequality, the distance between $\underline{y}$ and any of the 'incorrect' codewords is $\gtrsim N(\delta_{\rm GV}(R)-p)$. For $p < \delta_{\rm GV}(R)/2$ this is with high probability larger than the distance from $\underline{x}^{(0)}$.

The above argument implies that, for $p < \delta_{\rm GV}(R)/2$, the expected block error rate of a random code from Shannon's ensemble vanishes as $N \to \infty$. Notice that the channel coding theorem promises instead vanishing error probability whenever $R < 1 - \mathfrak{h}_2(p)$, that is (for $p < 1/2$) $p < \delta_{\rm GV}(R)$. The factor 2 of discrepancy can be recovered through a more careful argument.

Without entering into details, it is interesting to understand the basic reason for the discrepancy between the Shannon Theorem and the above argument. This is related to the geometry of high dimensional spaces. Let us assume for simplicity that the minimum distance between *any two* codewords in $\mathfrak{C}$ is at least $N(\delta_{\rm GV}(R)-\varepsilon)$. In a given random code, this is the case for most codeword pairs. We can then eliminate the pairs that do not satisfy this constraint, thus modifying the code rate in a negligible way (this procedure is called *expurgation*). The resulting code will have *minimum distance* (the minimum distance among any two codewords in $\mathfrak{C}$) $d(\mathfrak{C}) \approx N\delta_{\rm GV}(R)$.

Imagine that we use such a code to communicate through a BSC and that exactly $n$ bits are flipped. By the triangular inequality, as long as $n < d(\mathfrak{C})/2$, the word MAP decoder will recover the transmitted message for *all* error patterns. If on the other hand $n > d(\mathfrak{C})/2$, there are error patterns involving $n$ bits such that the word-MAP decoder does not return the transmitted codeword. If for instance there exists a single codeword $\underline{x}^{(1)}$ at distance $d(\mathfrak{C}) = 2n-1$ from $\underline{x}^{(0)}$, any pattern involving $n$ out of the $2n-1$ such that $x_i^{(0)} \neq x_i^{(1)}$, will induce a decoding error. However, it might well be that *most* error patterns with the same number of errors can be corrected.

Shannon's Theorem points out that this is indeed the case until the number of bits flipped by the channel is roughly equal to the minimum distance $d(\mathfrak{C})$.

# 3 Sparse Graph Codes

Shannon's Theorem provides a randomized construction to find a code with 'essentially optimal' rate vs error probability tradeoff. In practice, however, one cannot use random codes for communications. Just storing the code $\mathfrak{C}$ requires a memory which grows exponentially in the blocklength. In the same vein the optimal decoding procedure requires an exponentially increasing effort. On the other hand, we can not use very short codes since their performance is not very good. To see this assume that we transmit over the BSC with parameter $p$. If the blocklength is $N$ then the standard deviation of the number of errors contained in a block is $\sqrt{Np(1-p)}$. Unless this quantity is very small compared to $Np$ we have to either over-provision the error correcting capability of the code so as to deal with the occasionally large number of errors, waisting transmission rate most of the time, or we dimension the code for the typical case, but then we will not be able to decode when the number of errors is larger than the average. This means that short codes are either inefficient or unreliable (or both).

The general strategy for tackling this problem is to introduce more structure in the code definition, and to hope that such structure can be exploited for the encoding and the decoding. In the next section we shall describe a way of introducing structure that, while preserving Shannon's idea of random codes, opens the way to efficient encoding/decoding.

There are two main ingredients that make *modern coding* work and the two are tightly connected. The first important ingredient is to use codes which can be described by *local* constraints only. The second ingredient is to use a local algorithm instead of an high complexity global one (namely symbol MAP or word MAP decoding). In this section we describe the first component.

## 3.1 Linear Codes

One of the simplest forms of structure consists in requiring $\mathfrak{C}$ to be a linear subspace of $\{0,1\}^N$. One speaks then of a *linear code*. For specifying such a code it is not necessary to list all the codewords. In

fact, any linear space can be seen as the kernel of a matrix:

$$\mathfrak{C} = \left\{ \underline{x} \in \{0,1\}^N : \ \mathbb{H}\underline{x} = \underline{0} \right\} , \tag{3.1}$$

where the matrix vector multiplication is assumed to be performed modulo 2. The matrix $\mathbb{H}$ is called the *parity-check matrix*. It has $N$ columns and we let $M < N$ denote its number of rows. Without loss of generality we can assume $\mathbb{H}$ to have maximum rank $M$. As a consequence, $\mathfrak{C}$ is a linear space of dimension $N - M$. The rate of $\mathfrak{C}$ is

$$R = 1 - \frac{M}{N} . \tag{3.2}$$

The $a$-th line in $\mathbb{H}\underline{x} = \underline{0}$ has the form (here and below $\oplus$ denotes modulo 2 addition)

$$x_{i_1(a)} \oplus \cdots \oplus x_{i_k(a)} = 0. \tag{3.3}$$

It is called a *parity check*.

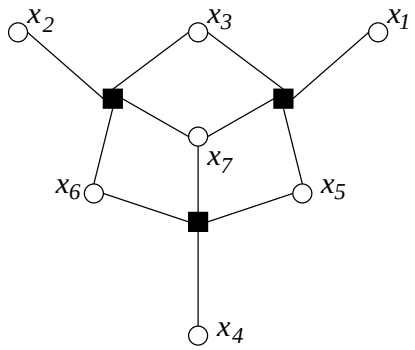


Figure 5: Factor graph for the parity-check matrix (3.4).

The parity-check matrix is conveniently represented through a *factor graph* (also called *Tanner graph*). This is a bipartite graph including two types of nodes: $M$ function nodes (corresponding to the rows of $\mathbb{H}$, or the parity-check equations) and $N$ variable nodes (for the columns of $\mathbb{H}$, or the variables). Edges are drawn whenever the corresponding entry in $\mathbb{H}$ is non-vanishing.

Example 4: In Fig. 5 we draw the factor graph corresponding to the parity-check matrix (here $N = 7$, $M = 3$)

$$\mathbb{H} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} . \tag{3.4}$$

In the following we shall use indices $i, j, \ldots$ for variable nodes and $a, b, \ldots$ for check nodes. We shall further denote by $\partial i$ (respectively, $\partial a$) the set of nodes that are adjacent to variable node $i$ (to factor node $a$).

Remarkably, introducing the linear space structure does not deteriorate the performances of the resulting code. Let us introduce Shannon's *parity-check ensemble*: it is defined by letting the parity-check matrix $\mathbb{H}$ be a uniformly random matrix with the prescribed dimensions. Explicitly, each of the $NM$ entries $H_{ai}$ is an independent Bernoulli random variable of mean $1/2$. Probabilistic arguments similar to the ones for the random code ensemble can be developed for the random parity-check ensemble.

The conclusion is that random codes from this ensemble allow to communicate with arbitrarily small block error probability at any rate $R < \mathtt{C}(Q)$, where $\mathtt{C}(Q)$ is the capacity of the given BMS channel.

Unfortunately, linearity is not sufficient to guarantee that a code admits a low-complexity decoding algorithm. In particular, the algorithm which we discuss in the sequel works well only for codes that can be represented by a sparse parity-check matrix $\mathbb{H}$ (i.e. a parity check matrix with $O(N)$ non-vanishing entries). Notice that a given code $\mathfrak{C}$ has more than one representation of the form (3.1). A priori one could hope that, given a uniformly random matrix $\mathbb{H}$, a new matrix $\mathbb{H}'$ could be built such that $\mathbb{H}'$ is sparse and that its null space coincides with the one of $\mathbb{H}$. This would provide a sparse representation of $\mathfrak{C}$. Unfortunately, this is the case only for a vanishing fraction of matrices $\mathbb{H}$, as shown by the exercise below.

Exercise 2: Consider a linear code $\mathfrak{C}$, with blocklength $N$, and dimension $N - M$ (as a linear space). Prove the following sequence of arguments.

(*i*) The total number of binary $N \times M$ parity-check matrices is $2^{NM}$.

(*ii*) Each code $\mathfrak{C}$ has $2^{\binom{M}{2}} \prod_{i=1}^{M} \left(2^i - 1\right)$ distinct $N \times M$ parity-check matrices $\mathbb{H}$.

(*iii*) The number of such matrices with at most $aN$ non-zero entries is $\sum_{i=0}^{aN} \binom{NM}{i} \leq 2^{NM\mathfrak{h}_2(a/(N-M))}$.

(*iv*) Conclude from the above that, for any given $a$, the fraction of parity-check matrices $\mathbb{H}$ that admit a sparse representation in terms of a matrix $\mathbb{H}'$ with at most $aN$ ones, is of order $e^{-N\gamma}$ for some $\gamma > 0$.

With an abuse of language in the following we shall sometimes use the term 'code' to denote a pair code/parity-check matrix.

## 3.2 Low-Density Parity-Check Codes

Further structure can be introduced by restricting the ensemble of parity-check matrices. Low-density parity-check (LDPC) codes are codes that have at least one sparse parity-check matrix.

Rather than considering the most general case let us limit ourselves to a particularly simple family of LDPC ensembles, originally introduced by Robert Gallager [19]. We call them 'regular' ensembles. An element in this family is characterized by the blocklength $N$ and two integer numbers $k$ and $l$, with $k > l$. We shall therefore refer to it as the $(k, l)$ regular ensemble). In order to construct a random Tanner graph from this ensemble, one proceeds as follows:

1. Draw $N$ variable nodes, each attached to $l$ half-edges and $M = Nl/k$ (we neglect here the possibility of $Nl/k$ not being an integer) check nodes, each with $k$ half edges.
2. Use an arbitrary convention to label the half edges form 1 to $Nl$, both on the variable node side as well as the check node side (note that this requires that $Mk = Nl$).
3. Choose a permutation $\pi$ uniformly at random among all permutations over $Nl$ objects, and connect half edges accordingly.

Notice that the above procedure may give rise to multiple edges. Typically there will be $O(1)$ multiple edges in a graph constructed as described. These can be eliminated easily without effecting the performance substantially. From the analytical point of view, a simple choice consists in eliminating all the edges $(i, a)$ if $(i, a)$ occurs an even number of times, and replacing them by a single occurrence $(i, a)$ if it occurs an odd number of times.

Neglecting multiple occurrences (and the way to resolve them), the parity-check matrix corresponding to the graph constructed in this way does include $k$ ones per row and $l$ ones per column. In the sequel we will keep $l$ and $k$ fixed and consider the behavior of the ensemble as $N \to \infty$. This implies that the matrix has only $O(N)$ non-vanishing entries. The matrix is *sparse*.

For practical purposes it is important to maximize the rate at which such codes enable one to communicate with vanishing error probability. To achieve this goal, several more complex ensembles have been introduced. As an example, one simple idea is to consider a generic row/column weight distribution (the weight being the number of non-zero elements), cf. Fig. 6 for an illustration. Such ensembles are usually referred to as 'irregular', and were introduced in [27].

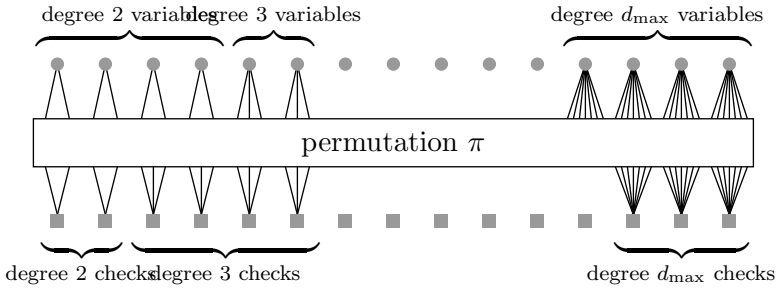


Figure 6: Factor graph of an irregular LDPC code. Variable nodes and function nodes can have any degree between 2 and $d_{\max}$. Half edges on the two sides are joined through a uniformly random permutation.

### 3.3 Weight Enumerator

As we saw in Section 2.5, the reason of the good performance of Shannon ensemble (having vanishing block error probability at rates arbitrarily close to the capacity), can be traced back to its minimum distance properties. This is indeed only a partial explanation (as we saw errors could be corrected well beyond half its minimum distance). It is nevertheless instructive and useful to understand the geometrical structure (and in particular the minimum distance properties) of typical codes from the LDPC ensembles defined above.

Let us start by noticing that, for linear codes, the distance enumerator does not depend upon the reference codeword. This is a straightforward consequence of the observation that, for any $\underline{x}^{(0)} \in \mathfrak{C}$ the set $\underline{x}^{(0)} \oplus \mathfrak{C} \equiv \{\underline{x}^{(0)} \oplus \underline{x} : \underline{x} \in \mathfrak{C}\}$ coincides with $\mathfrak{C}$. We are therefore led to consider the distance enumerator with respect to the all-zero codeword $\underline{0}$. This is also referred to as the *weight enumerator*,

$$\mathcal{N}(w) = \#\{\,\underline{x} \in \mathfrak{C} : \;\; w(\underline{x}) = w\,\}\;, \tag{3.5}$$

where $w(\underline{x}) = d(\underline{x}, \underline{0})$ is the number of non-zero entries in $\underline{x}$.

Let us compute the expected weight enumerator $\overline{\mathcal{N}}(w) \equiv \mathbb{E}\mathcal{N}(w)$. The final result is

$$\overline{\mathcal{N}}(w) = \frac{(lw)!(F-lw)!}{F!}\binom{N}{w}\,\mathsf{coeff}[q_k(z)^M, z^{lw}]\,. \tag{3.6}$$

Here, $F = Nl = Mk$ denotes the number of edges in the Tanner graph, $q_k(z) \equiv \frac{1}{2}[(1+z)^k + (1-z)^k]$, and, given a polynomial $p(z)$ and an integer $n$, $\mathsf{coeff}[p(z), z^n]$ denotes the coefficient of $z^n$ in the polynomial $p(z)$.

We shall now prove Eq. (3.6). Let $\underline{x} \in \{0,1\}^N$ be a binary word of length $N$ and weight $w$. Notice that $\mathbb{H}\underline{x} = 0$ if and only if the corresponding factor graph has the following property. Consider all variable nodes $i$ such that $x_i = 1$, and color in red all edges incident on these nodes. Color in blue all the other edges. Then all the check nodes must have an even number of incident red edges. A little thought shows that $\overline{\mathcal{N}}(w)$ is the number of 'colored' factor graphs having this property, divided by the total number of factor graphs in the ensemble.

A valid colored graph must have $wl$ red edges. It can be constructed as follows. First choose $w$ variable nodes. This can be done in $\binom{N}{w}$ ways. Assign to each node in this set $l$ red sockets, and to each node outside the set $l$ blue sockets. Then, for each of the $M$ function nodes, color in red an even subset of its sockets in such a way that the total number of red sockets is $E = wl$. The number of ways of doing this is[8] $\mathsf{coeff}[q_k(z)^M, z^{lw}]$. Finally we join the variable node and check node sockets in such a way that colors are matched. There are $(lw)!(F-lw)!$ such matchings out of the total number of $F!$ corresponding to different elements in the ensemble.

[8]This is a standard generating function calculation, and is explained in Appendix A.

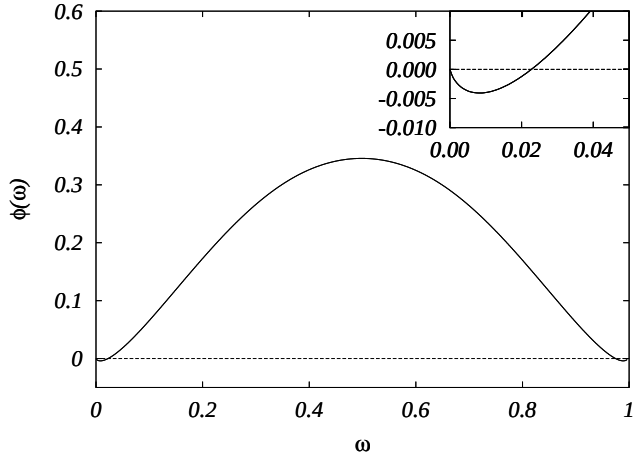


Figure 7: Logarithm of the expected weight enumerator for the $(3, 6)$ ensemble in the large blocklength limit. Inset: small weight region. Notice that $\phi(\omega) < 0$ for $\omega < \omega_* \approx 0.02$: besides the 'all-zero' word there is no codeword of weight smaller than $N\omega_*$ in the code with high probability.

Let us compute the exponential growth rate $\phi(\omega)$ of $\overline{\mathcal{N}}(w)$. This is defined by

$$\overline{\mathcal{N}}(w = N\omega) \doteq e^{N\phi(\omega)} \, . \tag{3.7}$$

In order to estimate the leading exponential behavior of Eq. (3.6), we set $w = N\omega$ and estimate the $\mathsf{coeff}[\dots, \dots]$ term using the Cauchy Theorem,

$$\mathsf{coeff}\left[q_k(z)^M, z^{wl}\right] = \oint \frac{q_k(z)^M}{z^{lw+1}} \frac{\mathrm{d}z}{2\pi i} = \oint \exp\left\{N[\left[\frac{l}{k}\log q_k(z) - l\omega \log z\right]\right\} \frac{\mathrm{d}z}{2\pi i} \, . \tag{3.8}$$

Here the integral runs over any path encircling the origin in the complex $z$ plane. Evaluating the integral using the saddle point method we finally get $\overline{\mathcal{N}}(w) \doteq e^{N\phi}$, where

$$\phi(\omega) \equiv (1-l)\mathfrak{h}(\omega) + \frac{l}{k}\log q_k(z) - \omega l \log z \, , \tag{3.9}$$

and $z$ is a solution of the saddle point equation

$$\omega = \frac{z}{k}\frac{q_k'(z)}{q_k(z)} \, . \tag{3.10}$$

The typical result of such a computation is shown in Fig. 7. As can be seen, there exists $\omega_* > 0$ such that $\phi(\omega) < 0$ for $\omega \in (0, \omega_*)$. This implies that a typical code from this ensemble will not have any codeword of weight between 0 and $N(\omega_* - \varepsilon)$. By linearity the minimum distance of the code is at least $\approx N\omega_*$. This implies in particular that such codes can correct any error pattern over the binary symmetric channel of weight $\omega_*/2$ or less.

Notice that $\phi(\omega)$ is an 'annealed average', in the terminology of disordered systems. As such, it can be dominated by rare instances in the ensemble. On the other hand, since $\log \mathcal{N}_N(N\omega) = \Theta(N)$ is an 'extensive' quantity, we expect it to be *self averaging* in the language of statistical physics. In mathematics terms one says that it should *concentrate in probability*. Formally, this means that there exists a function $\Phi_N(\omega)$ that is non-random (i.e., does not depend upon the code) and such that

$$\lim_{N\to\infty} \mathbb{P}\left\{|\log \mathcal{N}_N(N\omega) - \Phi_N(\omega)| \geq N\delta\right\} = 0 \, . \tag{3.11}$$

Further we expect that $\Phi_N(\omega) = N\phi_{\rm q}(\omega) + o(N)$ as $N \to \infty$. Despite being rather fundamental, both these statements are open conjectures.

The coefficient $\phi_{\rm q}(\omega)$ is the growth rate of the weight enumerator for typical codes in the ensembles. In statistical mechanics terms, it is a 'quenched' free energy (or rather, entropy). By Jensen inequality $\phi_{\rm q}(\omega) \le \phi(\omega)$. A statistical mechanics calculation reveals that the inequality is strict for general (irregular) ensembles. On the other hand, for regular ensembles as the ones considered here, $\phi_{\rm q}(\omega) = \phi(\omega)$: the annealed calculation yields the correct exponential rate. This claim has been supported rigorously by the results of [47, 3, 31].

Let us finally comment on the relation between distance enumerator and the Franz-Parisi potential [18], introduced in the study of glassy systems. In this context the potential is used to probe the structure of the Boltzmann measure. One considers a system with energy function $E(x)$, a reference configuration $x_0$ and some notion of distance between configurations $d(x, x')$. The constrained partition function is then defined as

$$Z(x_0, w) = \int e^{-E(x)}\, \delta(d(x_0, x) - w)\, {\rm d}x\,. \tag{3.12}$$

One then defines the potential $\Phi_N(\omega)$ as the typical value of $\log Z(x_0, w)$ when $x_0$ is a random configuration with the same Boltzmann distribution and $w = N\omega$. Self averaging is expected to hold here too:

$$\lim_{N\to\infty} \mathbb{P}_{x_0}\{\, |\log Z(x_0, N\omega) - \Phi_N(\omega)| \ge N\delta\,\} = 0\,. \tag{3.13}$$

Here $N$ may denote the number of particles or the volume of the system and $\mathbb{P}_{x_0}\{\cdots\}$ indicates probability with respect to $x_0$ distributed with the Boltzmann measure for the energy function $E(x_0)$.

It is clear that the two ideas are strictly related and can be generalized to any joint distribution of $N$ variables $(x_1, \dots, x_N)$. In both cases the structure of such a distribution is probed by picking a reference configuration and restricting the measure to its neighborhood.

To be more specific, the weight enumerator can be seen as a special case of the Franz-Parisi potential. It is sufficient to take as Boltzmann distribution the uniform measure over codewords of a linear code $\mathfrak{C}$. In other words, let the configurations be binary strings of length $N$, and set $E(\underline{x}) = 0$ if $\underline{x} \in \mathfrak{C}$, and $= \infty$ otherwise. Then the restricted partition function is just the distance enumerator with respect to the reference codeword, which indeed does not depend on it.

# 4 The Decoding Problem for Sparse Graph Codes

As we have already seen, MAP decoding requires computing either marginals or the mode of the conditional distribution of $\underline{x}$ being the channel input given output $\underline{y}$. In the case of LDPC codes the posterior probability distribution factorizes according to underlying factor graph $G$:

$$\mu_{\mathfrak{C},y}(\underline{x}) = \frac{1}{Z(\mathfrak{C}, y)} \prod_{i=1}^{N} Q(y_i|x_i) \prod_{a=1}^{M} \mathbb{I}(x_{i_1(a)} \oplus \cdots \oplus x_{i_k(a)} = \mathtt{0})\,. \tag{4.1}$$

Here $(i_1(a), \dots, i_k(a))$ denotes the set of variable indices involved in the $a$-th parity check (i.e., the non-zero entries in the $a$-th row of the parity-check matrix $\mathbb{H}$). In the language of spin models, the terms $Q(y_i|x_i)$ correspond to an external random field. The factors $\mathbb{I}(x_{i_1(a)} \oplus \cdots \oplus x_{i_k(a)} = \mathtt{0})$ can instead be regarded as hard core $k$-spins interactions. Under the mapping $\sigma_i = (-1)^{x_i}$, such interactions depend on the spins through the product $\sigma_{i_1(a)} \cdots \sigma_{i_k(a)}$. The model (4.1) maps therefore onto a $k$-spin model with random field.

For MAP decoding, minimum distance properties of the code play a crucial role in determining the performances. We investigated such properties in the previous section. Unfortunately, there is no known way of implementing MAP decoding efficiently. In this section we discuss two decoding algorithms that

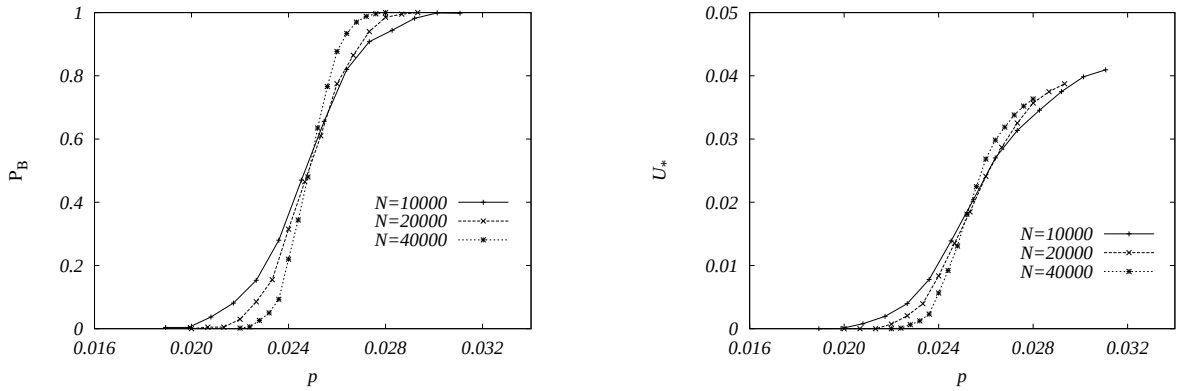


Figure 8: Numerical simulations of bit-flipping decoding of random codes from the $(5, 10)$ regular ensemble. On the left: block error rate achieved by this scheme. On the right: fraction of unsatisfied parity checks in the word found by the algorithm.

exploit the sparseness of the factor graph to achieve efficient decoding. Although such strategies are sub-optimal with respect to word (or symbol) MAP decoding, the graphical structure can itself be optimized, leading to state-of-the-art performances.

After briefly discussing bit-flipping decoding, most of this section will be devoted to message passing that is the approach most used in practice. Remarkably, both bit flipping as well as message passing are closely related to statistical mechanics.

## 4.1 Bit Flipping

For the sake of simplicity, let us assume that communication takes place over a binary symmetric channel. We receive the message $\underline{y} \in \{0, 1\}^N$ and try to find the transmitted codeword $\underline{x}$ as follows:

Bit-flipping decoder

0. Set $\underline{x}(0) = \underline{y}$.
1. Find a bit belonging to more unsatisfied than satisfied parity checks.
2. If such a bit exists, flip it: $x_i(t+1) = x_i(t) \oplus 1$. Keep the other bits: $x_j(t+1) = x_j(t)$ for all $j \neq i$.
   If there is no such bit, return $\underline{x}(t)$ and halt.
3. Repeat steps 1 and 2.

The bit to be flipped is usually chosen uniformly at random among the ones satisfying the condition at step 1. However this is irrelevant for the analysis below.

In order to monitor the bit-flipping algorithm, it is useful to introduce the function:

$$U(t) \equiv \#\{ \text{ parity-check equations not satisfied by } \underline{x}(t) \ \} \ . \tag{4.2}$$

This is a non-negative integer, and if $U(t) = 0$ the algorithm is halted and it outputs $\underline{x}(t)$. Furthermore, $U(t)$ cannot be larger than the number of parity checks $M$ and decreases (by at least one) at each cycle. Therefore, the algorithm complexity is $O(N)$ (this is a commonly regarded as the ultimate goal for many communication problems).

It remains to be seen if the output of the bit-flipping algorithm is related to the transmitted codeword. In Fig. 8 we present the results of a numerical experiment. We considered the $(5, 10)$ regular ensemble and generated about 1000 random code and channel realizations for each value of the noise level $p$ in some

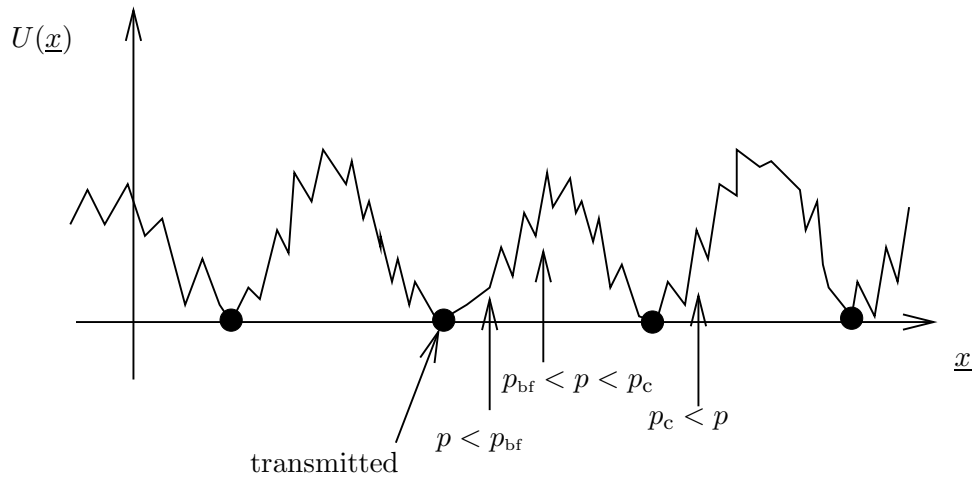


Figure 9: Sketch of the cost function $U(\underline{x})$ (number of unsatisfied parity checks) for a typical random LDPC code. Filled circles correspond to codewords, and arrows to received messages in various possible regimes.

mesh. Then, we applied the above algorithm and traced the fraction of successfully decoded blocks, as well as the residual energy $U_* = U(t_*)$, where $t_*$ is the total number of iterations of the algorithm. The data suggests that bit-flipping is able to overcome a finite noise level: it recovers the original message with high probability when less than about 2.5% of the bits are corrupted by the channel. Furthermore, the curves for the block error probability $\mathrm{P_B^{bf}}$ under bit-flipping decoding become steeper and steeper as the system size is increased. It is natural to conjecture that asymptotically, a phase transition takes place at a well defined noise level $p_{\rm bf}$: $\mathrm{P_B^{bf}} \to 0$ for $p < p_{\rm bf}$ and $\mathrm{P_B^{bf}} \to 1$ for $p > p_{\rm bf}$. Numerically $p_{\rm bf} = 0.025 \pm 0.005$.

This threshold can be compared with the one for word MAP decoding, that we will call $p_{\rm c}$: The bounds in [60] state that $0.108188 \le p_{\rm c} \le 0.109161$ for the $(5, 10)$ ensemble, while a statistical mechanics calculation yields $p_{\rm c} \approx 0.1091$. Bit-flipping is significantly sub-optimal, but it is still surprisingly good, given the extreme simplicity of the algorithm.

These numerical findings can be confirmed rigorously [55].

**Theorem 4.1.** *Consider a regular $(l, k)$ LDPC ensemble and let $\mathfrak{C}$ be chosen uniformly at random from the ensemble. If $l \ge 5$ then there exists $\varepsilon > 0$ such that, with high probability,* `Bit-flipping` *is able to correct any pattern of at most $N\varepsilon$ errors produced by a binary symmetric channel.*

Given a generic word $\underline{x}$ (i.e., a length $N$ binary string that is not necessarily a codeword), let us denote, with a slight abuse of notation, by $U(\underline{x})$ the number of parity-check equations that are not satisfied by $\underline{x}$. The above result, together with the weight enumerator calculation in the previous section, suggests the following picture of the function $U(\underline{x})$. If $\underline{x}^{(0)} \in \mathfrak{C}$, than $U(\underline{x}^{(0)}) = 0$. Moving away from $\underline{x}^{(0)}$, $U(\underline{x})$ will become strictly positive. However as long as $d(\underline{x}^{(0)}, \underline{x})$ is small enough, $U(\underline{x})$ does not have any local minimum distinct from $\underline{x}^{(0)}$. A greedy procedure with a starting point within such a Hamming radius is able to reconstruct $\underline{x}^{(0)}$. As we move further away, $U(\underline{x})$ stays positive (no other codewords are encountered) but local minima start to appear. Bit flipping gets trapped in such minima. Finally, for $d(\underline{x}^{(0)}, \underline{x}) \ge N\omega_*$ new codewords, i.e., minima with $U(\underline{x}) = 0$, are encountered.

## 4.2 Message Passing

Message-passing algorithms are iterative and have low complexity. Unlike the bit-flipping procedure in the previous section, the basic variables are now associated to directed edges in the factor graph. More precisely, for each edge $(i, a)$ (corresponding to a non-zero entry in the parity-check matrix at row $a$ and column $i$), we introduce two messages $\nu_{i\to a}$ and $\widehat{\nu}_{a\to i}$. Messages are elements of some set (the message

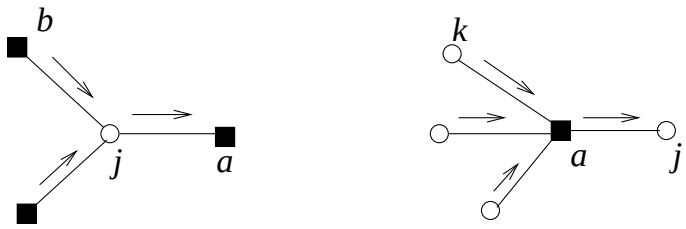


Figure 10: Graphical representation of message passing updates.

alphabet) that we shall denote by $\mathsf{M}$. Depending on the specific algorithm, $\mathsf{M}$ can have finite cardinality, or be infinite, for instance $\mathsf{M} = \mathbb{R}$. Since the algorithm is iterative, it is convenient to introduce a time index $t = 0, 1, 2, \dots$ and label the messages with the time at which they are updated: $\nu_{i\to a}^{(t)}$ and $\widehat{\nu}_{a\to i}^{(t)}$ (but we will sometimes drop the label below).

The defining property of message-passing algorithms is that the message flowing from node $u$ to $v$ at a given time is a function of messages entering $u$ from nodes $w$ distinct from $v$ at the previous time step. Formally, the algorithm is defined in terms of two sets of functions $\Phi_{i\to a}(\,\cdot\,)$, $\Psi_{a\to i}(\,\cdot\,)$, that define the update operations at variable and function nodes as follows

$$\nu_{i\to a}^{(t+1)} = \Phi_{i\to a}(\{\widehat{\nu}_{b\to i}^{(t)}; b \in \partial i\backslash a\}; y_i)\,, \qquad \widehat{\nu}_{a\to i}^{(t)} = \Psi_{a\to i}(\{\nu_{j\to a}^{(t)}; j \in \partial a\backslash i\})\,. \tag{4.3}$$

Notice that messages are updated in parallel and that the time counter is incremented only at variable nodes. Alternative scheduling schemes can be considered but we will stick to this for the sake of simplicity. After a pre-established number of iterations, the transmitted bits are estimated using *all* the messages incoming at the corresponding nodes. More precisely, the estimate at function $i$ is defined through a new function

$$\widehat{x}_i^{(t)}(\underline{y}) = \Phi_i(\{\widehat{\nu}_{b\to i}^{(t)}; b \in \partial i\}; y_i)\,. \tag{4.4}$$

A graphical representation of message passing updates is provided in Fig. 10.

A specific message-passing algorithm requires the following features to be specified:

1. The message alphabet $\mathsf{M}$.
2. The initialization $\{\nu_{i\to a}^{(0)}\}$, $\{\widehat{\nu}_{i\to a}^{(0)}\}$.
3. The update functions $\{\Phi_{i\to a}(\,\cdot\,)\}$, $\{\Psi_{a\to i}(\,\cdot\,)\}$.
4. The final estimate functions $\{\Phi_i(\,\cdot\,)\}$.

The most prominent instance of a message-passing algorithm is the Belief Propagation (BP) algorithm. In this case the messages $\nu_{i\to a}^{(t)}(x_i)$ and $\widehat{\nu}_{a\to i}^{(t)}(x_i)$ are distributions over the bit variables $x_i \in \{0, 1\}$. The message $\widehat{\nu}_{a\to i}^{(t)}(x_i)$ is usually interpreted as the *a posteriori* distributions of the bit $x_i$ given the information coming from edge $a \to i$. Analogously, $\nu_{i\to a}(x_i)$ is interpreted as the *a posteriori* distribution of $x_i$, given all the information collected through edges distinct from $(a, i)$. Since the messages normalization (explicitly $\nu_{i\to a}(0) + \nu_{i\to a}(1) = 1$) can be enforced at any time, we shall neglect overall factors in writing down the relation between to messages (and correspondingly, we shall use the symbol $\propto$).

BP messages are updated according to the following rule, whose justification we will discuss in the next section

$$\nu_{i\to a}^{(t+1)}(x_i) \;\propto\; Q(y_i|x_i) \prod_{b\in\partial i\backslash a} \widehat{\nu}_{b\to i}^{(t)}(x_i)\,, \tag{4.5}$$

$$\widehat{\nu}_{a\to i}^{(t)}(x_i) \;\propto\; \sum_{\{x_j\}} \mathbb{I}(x_i \oplus x_{j_1} \oplus \cdots \oplus x_{j_{k-1}} = 0) \prod_{j\in\partial a\backslash i} \nu_{j\to a}^{(t)}(x_j)\,, \tag{4.6}$$

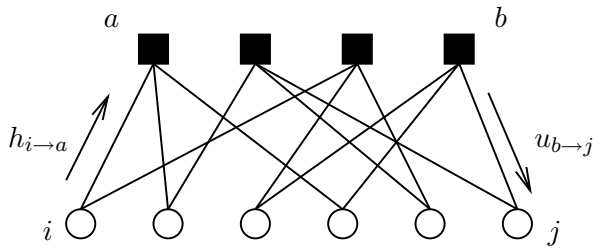


Figure 11: Factor graph of a regular LDPC code, and notation for the belief propagation messages.

where we used $(i, j_1, \dots, j_{k-1})$ to denote the neighborhood $\partial a$ of factor node $a$. After any number of iterations the single bit marginals can be estimated as follows

$$\nu_i^{(t+1)}(x_i) \;\propto\; Q(y_i|x_i) \prod_{b\in\partial i} \widehat{\nu}_{b\to i}^{(t)}(x_i)\,. \tag{4.7}$$

The corresponding MAP decision for bit $i$ (sometimes called 'hard decision', while $\nu_i(x_i)$ is the 'soft decision') is

$$\widehat{x}_i^{(t)} = \arg\max_{x_i}\; \nu_i^{(t)}(x_i)\,. \tag{4.8}$$

Notice that the above prescription is ill-defined when $\nu_i(\mathtt{0}) = \nu_i(\mathtt{1})$. It turns out that it is not really important which rule to use in this case. To preserve the $\mathtt{0}-\mathtt{1}$ symmetry, we shall assume that the decoder returns $\widehat{x}_i^{(t)} = \mathtt{0}$ or $= \mathtt{1}$ with equal probability.

Finally, as initial condition one usually takes $\widehat{\nu}_{a\to i}^{(-1)}(\,\cdot\,)$ to be the uniform distribution over $\{\mathtt{0},\mathtt{1}\}$ (explicitly $\widehat{\nu}_{a\to i}^{(-1)}(\mathtt{0}) = \widehat{\nu}_{a\to i}^{(-1)}(\mathtt{1}) = 1/2$).

Since for BP the messages are distributions over binary valued variables, they can be described by a single real number, that is often chosen to be the bit log-likelihood:[9]

$$h_{i\to a} = \frac{1}{2}\,\log\frac{\nu_{i\to a}(\mathtt{0})}{\nu_{i\to a}(\mathtt{1})}\,, \qquad u_{a\to i} = \frac{1}{2}\,\log\frac{\widehat{\nu}_{a\to i}(\mathtt{0})}{\widehat{\nu}_{a\to i}(\mathtt{1})}\,. \tag{4.9}$$

We refer to Fig. 11 for a pictorial representation of these notations. We further introduce the channel log-likelihoods

$$B_i = \frac{1}{2}\,\log\frac{Q(y_i|\mathtt{0})}{Q(y_i|\mathtt{1})}\,. \tag{4.10}$$

The BP update equations (4.5), (4.6) read in this notation

$$h_{i\to a}^{(t+1)} = B_i + \sum_{b\in\partial i\setminus a} u_{b\to i}^{(t)}\,, \qquad u_{a\to i}^{(t)} = \operatorname{atanh}\Big\{\prod_{j\in\partial a\setminus i} \tanh h_{j\to a}^{(t)}\Big\}\,. \tag{4.11}$$

In this language the standard message initialization would be $u_{a\to i}^{(-1)} = 0$. Finally, the overall log-likelihood at bit $i$ is obtained by combining *all* the incoming messages in agreement with Eq. (4.7). One thus gets the decision rule

$$\widehat{x}_i^{(t)} = \begin{cases} \mathtt{0} & \text{if } B_i + \sum_{b\in\partial i} u_{b\to i}^{(t)} > 0, \\ \mathtt{1} & \text{if } B_i + \sum_{b\in\partial i} u_{b\to i}^{(t)} < 0. \end{cases} \tag{4.12}$$

[9]The conventional definition of log-likelihoods does not include the factor 1/2. We introduce this factor here for uniformity with the statistical mechanics convention (the $h$'s and $u$'s being analogous to effective magnetic fields).

Notice that we did not commit to any special decision if $B_i + \sum_{b \in \partial i} u_{b \to i}^{(t)} = 0$. To keep complete symmetry we'll establish that the decoder returns 0 or 1 with equal probability in this case.

## 4.3 Correctness of Belief Propagation on Trees

The justification for the BP update equations (4.5), (4.6) lies in the observation that, whenever the underlying factor graph is a tree, the estimated marginal $\nu_i^{(t)}(x_i)$ converges after a finite number of iterations to the correct one $\mu_i(x_i)$. In particular, under the tree assumption, and for any $t$ sufficiently large, $\widehat{x}_i^{(t)}(\underline{y})$ coincides with the symbol MAP decision.

In order to prove this statement, consider a tree factor graph $G$. Given a couple of adjacent nodes $u, v$, denote by $G(u \to v)$ the subtree rooted at the directed edge $u \to v$ (this contains all that can be reached from $v$ through a non-reversing path whose first step is $v \to u$). If $i$ is a variable index and $a$ a parity-check index, let $\mu_{i \to a}(\,\cdot\,)$ be the measure over $\underline{x} = \{x_j : j \in G(i \to a)\}$, that is obtained by retaining in Eq. (4.1) only those terms that are related to nodes in $G(i \to a)$:

$$\mu_{i \to a}(\underline{x}) = \frac{1}{Z(i \to a)} \prod_{j \in G(i \to a)} Q(y_i | x_i) \prod_{b \in G(i \to a)} \mathbb{I}(x_{i_1(b)} \oplus \cdots \oplus x_{i_k(b)} = 0)\,. \tag{4.13}$$

The measure $\widehat{\mu}_{a \to i}(\,\cdot\,)$ is defined analogously for the subtree $G(a \to i)$. The marginals $\mu_{i \to a}(x_i)$ (respectively $\widehat{\mu}_{a \to i}(x_i)$) are easily seen to satisfy the recursions

$$\mu_{i \to a}(x_i) \;\propto\; Q(y_i|x_i) \prod_{b \in \partial i \setminus a} \widehat{\mu}_{b \to i}(x_i)\,, \tag{4.14}$$

$$\widehat{\mu}_{a \to i}(x_i) \;\propto\; \sum_{\{x_j\}} \mathbb{I}(x_i \oplus x_{j_1} \oplus \cdots \oplus x_{j_{k-1}} = 0) \prod_{j \in \partial a \setminus i} \mu_{j \to a}(x_j)\,, \tag{4.15}$$

which coincide, apart from the time index, with the BP recursion (4.5), (4.6). That such recursions converges to $\{\mu_{i \to a}(x_i), \widehat{\mu}_{a \to i}(x_i)\}$ follows by induction over the tree depth.

In statistical mechanics equations similar to (4.14), (4.15) are often written as recursions on the constrained partition function. They allow to solve exactly models on trees. However they have been often applied as mean-field approximation to statistical models on non-tree graphs. This is often referred to as the *Bethe-Peierls approximation* [8].

The Bethe approximation presents several advantages with respect to 'naive-mean field' [61] (that amounts to writing 'self-consistency' equations for expectations over single degrees of freedom). It retains correlations among degrees of freedom that interact directly, and is exact on some non-empty graph (trees). It is often asymptotically (in the large size limit) exact on locally tree-like graphs. Finally, it is quantitatively more accurate for non-tree like graphs and offers a much richer modeling palette.

Within the theory of disordered systems (especially, glass models on sparse random graphs), Eqs. (4.14) and (4.15) are also referred to as the *cavity equations*. With respect to Bethe-Peierls, the cavity approach includes a hierarchy of ('replica symmetry breaking') refinements of such equations that aim at capturing long range correlations [37]. This will briefly described in Section 5.

We should finally mention that several improvements over Bethe approximation have been developed within statistical physics. Among them, Kikuchi's cluster variational method [24] is worth mentioning since it motivated the development of a 'generalized belief propagation' algorithm, which spurred a lot of interest within the artificial intelligence community [61].

## 4.4 Density Evolution

Although BP converges to the exact marginals on tree graphs, this says little about its performances on practical codes such as the LDPC ensembles introduced in Section 3. Fortunately, a rather precise picture on the performance of LDPC ensembles can be derived in the large blocklength limit $N \to \infty$.

The basic reason for this is that the corresponding random factor graph is locally tree-like with high probability if we consider large blocklengths.

Before elaborating on this point, notice that the performance under BP decoding (e.g., the bit error rate) is independent on the transmitted codeword. For the sake of analysis, we shall hereafter assume that the all-zero codeword $\underline{0}$ has been transmitted.

Consider a factor graph $G$ and let $(i,a)$ be one of its edges. Consider the message $\nu_{i\to a}^{(t)}$ sent by the BP decoder in iteration $t$ along edge $(i,a)$. A considerable amount of information is contained in the distribution of $\nu_{i\to a}^{(t)}$ with respect to the channel realization, as well as in the analogous distribution for $\widehat{\nu}_{a\to i}^{(t)}$. To see this, note that under the all-zero codeword assumption, the bit error rate after $t$ iterations is given by

$$\mathrm{P}_{\mathrm{b}}^{(t)} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{P}\left\{\Phi_i(\{\widehat{\nu}_{b\to i}^{(t)}; b\in\partial i\}; y_i) \neq \mathsf{0}\right\} . \tag{4.16}$$

Therefore, if the messages $\widehat{\nu}_{b\to i}^{(t)}$ are independent, then the bit error probability is determined by the distribution of $\widehat{\nu}_{a\to i}^{(t)}$.

Rather than considering one particular graph (code) and a specific edge, it is much simpler to take the average over all edges and all graph realizations. We thus consider the distribution $\mathsf{a}_t^{(N)}(\,\cdot\,)$ of $\nu_{i\to a}^{(t)}$ with respect to the channel, the edges, *and* the graph realization. While this is still a quite difficult object to study rigorously, it is on the other hand possible to characterize its large blocklength limit $\mathsf{a}_t(\,\cdot\,) = \lim_N \mathsf{a}_t^{(N)}(\,\cdot\,)$. This distribution satisfies a simple recursion.

It is convenient to introduce the *directed neighborhood* of radius $r$ of the directed edge $i\to a$ in $G$, call it $\mathsf{B}_{i\to a}(r;G)$. This is defined as the subgraph of $F$ that includes all the variable nodes that can be reached from $i$ through a non-reversing path of length at most $r$, whose first step *is not* the edge $(i,a)$. It includes as well all the function nodes connected *only* to the above specified variable nodes. In Fig. 13 we reproduce an example of a directed neighborhood of radius $r=3$ (for illustrative purposes we also include the edge $(i,a)$) in a $(2,3)$ regular code.

If $F$ is the factor graph of a random code from the $(k,l)$ LDPC ensemble, then $\mathsf{B}_{i\to a}(r;F)$ is with high probability a depth-$r$ regular tree with degree $l$ at variable nodes and degree $k$ at check nodes (as in Fig. 13 where $l=2$ and $k=3$). The basic reason for this phenomenon is rather straightforward. Imagine to explore the neighborhood progressively, moving away from the root, in a breadth first fashion. At any finite radius $r$, about $c^r/N$ vertices have been visited (here $c=(k-1)(l-1)$). The vertices encountered at the next layer will be 'more or less' uniformly random among all the ones not visited so far. As a consequence they will be distinct with high probability, and $\mathsf{B}_{i\to a}(r+1;G)$ will be a tree as well. This argument breaks down when the probability that two of the $\Theta(c^r)$ new vertices coincide, that is for $c^{2r}=\Theta(N)$.[10] This is equivalent to $r\simeq \frac{1}{2}\log_c N$.

The skeptical reader is invited to solve the following exercise.

> Exercise 3: In order to illustrate the above statement, consider the example of a random code from the regular $(2,3)$ ensemble (each variable has degree 2 and each check has degree 3). The three possible radius-1 neighborhoods appearing in the associated factor graph are depicted in Fig. 14.
>
> (*a*) Show that the probability that a given edge $(i,a)$ has neighborhoods as in (B) or (C) is $O(1/N)$.
>
> (*b*) What changes for a generic radius $r$?

For illustrative reasons, we shall occasionally add a 'root edge' to $\mathsf{B}_{i\to a}(r;G)$, as for $i\to a$ in Fig. 13.

Now consider the message $\nu_{i\to a}^{(t)}$. This is a function of the factor graph $G$ and of the received message $\underline{y}$. However, a moment's thought shows that it will depend on $G$ only through its directed neighborhood $\mathsf{B}_{i\to a}(t+1;G)$, and only on the received symbols $y_j$, $j\in\mathsf{B}_{i\to a}(t;G)$.

[10] This is the famous birthday problem. The probability that two out of a party of $n$ peoples were born on the same day of the year, scales like $n^2/N$ for $n^2\ll N$ ($N$ is here the number of days in a year).

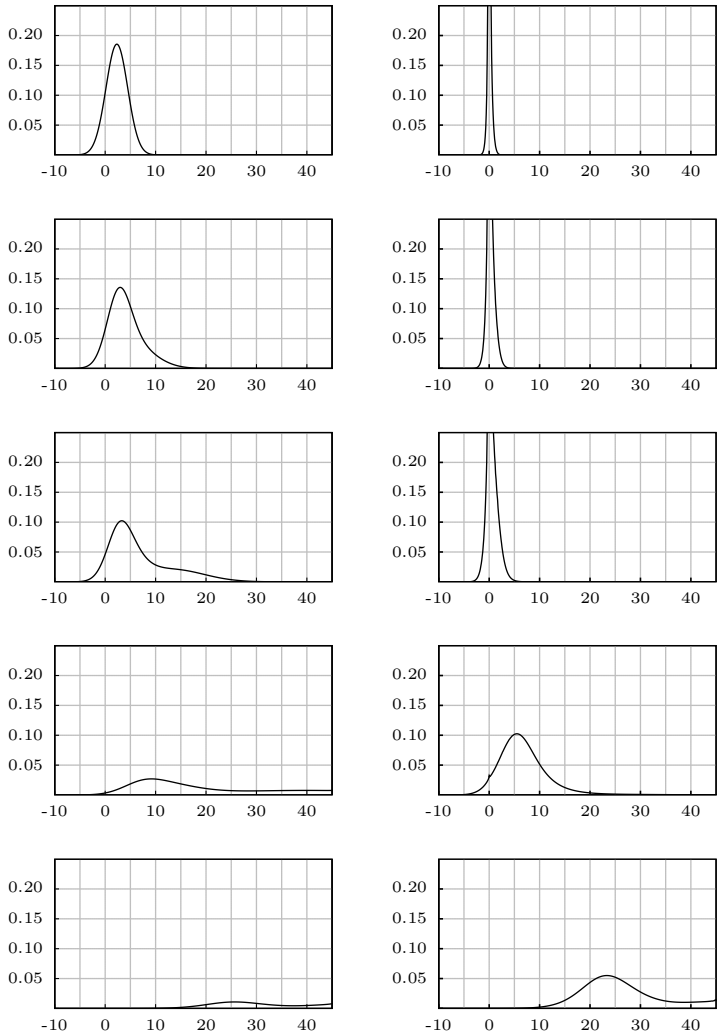


Figure 12: Evolution of the probability density functions of $h^{(t)}$ an $u^{(t+1)}$ for an irregular LDPC code used over a gaussian channel. From top to bottom $t = 0$, 5, 10, 50, and 140.

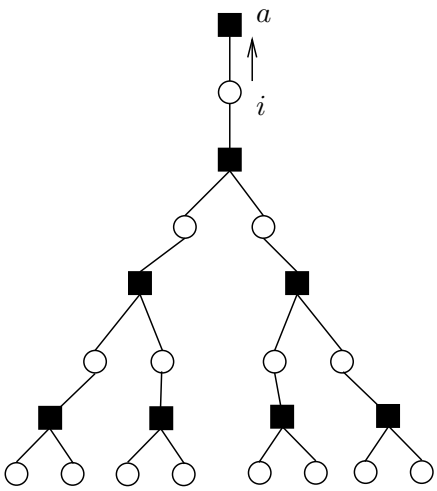


Figure 13: A radius 3 directed neighborhood $\mathsf{B}_{i\to a}(3;G)$.

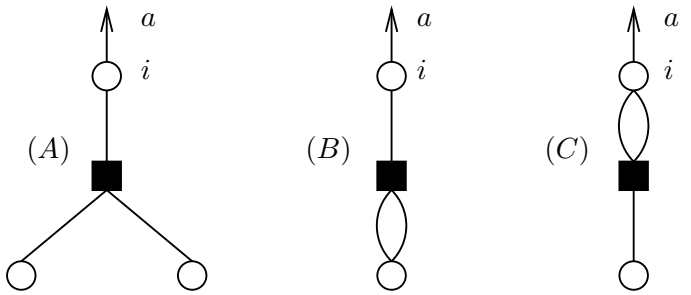


Figure 14: The three possible radius–1 directed neighborhoods in a random factor graph from the regular $(2,3)$ graph ensemble.

In view of the above discussion, let us consider the case in which $\mathsf{B}_{i\to a}(t+1;G)$ is a $(k,l)$-regular tree. We further assume that the received symbols $y_j$ are i.i.d. with distribution $Q(y|\mathsf{0})$, and that the update rules (4.3) do not depend on the edge we are considering (i.e., $\Phi_{i\to a}(\,\cdot\,)=\Phi(\,\cdot\,)$ and $\Psi_{i\to a}(\,\cdot\,)=\Psi(\,\cdot\,)$ independent of $i$, $a$).

Let $\nu^{(t)}$ be the message passed through the root edge of such a tree after $t$ BP iterations. Since the actual neighborhood $\mathsf{B}_{i\to a}(t+1;G)$ is with high probability a tree, $\nu^{(t)}_{i\to a}\stackrel{\mathrm{d}}{\to}\nu^{(t)}$ as $N\to\infty$. The symbol $\stackrel{\mathrm{d}}{\to}$ denotes convergence in distribution. In other words, for large blocklengths, the message distribution after $t$ iterations is asymptotically the same that we would have obtained if the graph were a tree.

Consider now a $(k,l)$-regular tree, and let $j\to b$ an edge directed towards the root, at distance $d$ from it. It is not hard to realize that the message passed through it after $r-d-1$ (or more) iterations is distributed as $\nu^{(r-d-1)}$. Furthermore, if $j_1\to b_1$ and $j_2\to b_2$ are both directed upwards and none belongs to the subtree rooted at the other one, then the corresponding messages are independent. Together with Eq. (4.3), these observation imply that

$$\nu^{(t+1)}\stackrel{\mathrm{d}}{=}\Phi(\widehat{\nu}^{(t)}_1,\dots,\widehat{\nu}^{(t)}_{l-1};y)\,,\qquad \widehat{\nu}^{(t)}\stackrel{\mathrm{d}}{=}\Psi(\nu^{(t)}_1,\dots,\nu^{(t)}_{k-1})\,. \tag{4.17}$$

Here $\widehat{\nu}^{(t)}_1,\dots,\widehat{\nu}^{(t)}_{l-1}$ are i.i.d. copies of $\widehat{\nu}^{(t)}$, and $\nu^{(t)}_1,\dots,\nu^{(t)}_{k-1}$ i.i.d. copies of $\nu^{(t)}$. Finally, $y$ is a received

symbol independent from the previous variables and distributed according to $Q(y|0)$.

Equations (4.17), or the sequence of distributions that they define, are usually referred to as *density evolution*. The name is motivated by the identification of the random variables with their densities (even if these do not necessarily exist). They should be parsed as follows (we are refer here to the first equation in (4.17); an analogous phrasing holds for the second): pick $l-1$ i.i.d. copies $\widehat{\nu}^{(t)}$ and $y$ with distribution $Q(y|0)$, compute $\Phi(\widehat{\nu}_1^{(t)},\dots,\widehat{\nu}_{l-1}^{(t)};y)$. The resulting quantity will have distribution $\nu^{(t+1)}$. Because of this description, they are also called 'recursive distributional equations'.

Until this point we considered a generic message passing procedure. If we specialize to BP decoding, we can use the parametrization of messages in terms of log-likelihood ratios, cf. Eq. (4.9), and use the above arguments to characterize the limit random variables $h^{(t)}$ and $u^{(t)}$. The update rules (4.11) then imply

$$h^{(t+1)} \stackrel{\mathrm{d}}{=} B + u_1^{(t)} + \cdots + u_{l-1}^{(t)}\,, \qquad u^{(t)} \stackrel{\mathrm{d}}{=} \operatorname{atanh}\Big\{\tanh h_1^{(t)} \cdots \tanh h_{k-1}^{(t)}\Big\}\,. \tag{4.18}$$

Here $u_1^{(t)},\dots,u_{l-1}^{(t)}$ are i.i.d. copies of $u^{(t)}$, $h_1^{(t)},\dots,h_{k-1}^{(t)}$ are i.i.d. copies of $h^{(t)}$, and $B = \frac{1}{2}\log\frac{Q(y|0)}{Q(y|1)}$, where $y$ is independently distributed according to $Q(y|0)$. It is understood that the recursion is initiated with $u^{(-1)} = 0$.

Physicists often write distributional recursions explicitly in terms of densities. For instance, the first of the equations above reads

$$\mathsf{a}_{t+1}(h) = \int \prod_{b=1}^{l-1} \mathrm{d}\widehat{\mathsf{a}}_t(u_b)\,\mathrm{d}\mathsf{p}(B) \quad \delta\left(h - B - \sum_{b=1}^{l-1} u_b\right)\,, \tag{4.19}$$

where $\widehat{\mathsf{a}}_t(\,\cdot\,)$ denotes the density of $u^{(t)}$, and $\mathsf{p}(\,\cdot\,)$ the density of $B$. We refer to Fig. 12 for an illustration of how the densities $\mathsf{a}_t(\,\cdot\,)$, $\widehat{\mathsf{a}}_t(\,\cdot\,)$ evolve during the decoding process.

In order to stress the importance of density evolution notice that, for any continuous function $f(x)$,

$$\lim_{N\to\infty} \mathbb{E}\Big\{\frac{1}{N}\sum_{i=1}^{N} f(h_{i\to a}^{(t)})\Big\} = \mathbb{E}\{f(h^{(t)})\}\,, \tag{4.20}$$

where the expectation is taken with respect to the code ensemble. Similar expressions can be obtained for functions of several messages (and are particularly simple when such message are asymptotically independent). In particular[11], if we let $\mathrm{P}_{\mathrm{b}}^{(N,t)}$ be the expected (over an LDPC ensemble) bit error rate for the decoding rule (4.12), and let $\mathrm{P}_{\mathrm{b}}^{(t)} = \lim_{N\to\infty}\mathrm{P}_{\mathrm{b}}^{(N,t)}$ be its large blocklength limit. Then

$$\mathrm{P}_{\mathrm{b}}^{(t)} = \mathbb{P}\big\{B + h_1^{(t)} + \cdots + h_l^{(t)} < 0\big\} + \frac{1}{2}\,\mathbb{P}\big\{B + h_1^{(t)} + \cdots + h_l^{(t)} = 0\big\}\,, \tag{4.21}$$

where $h_1^{(t)},\dots,h_l^{(t)}$ are i.i.d. copies of $h^{(t)}$.

## 4.5 The Belief Propagation Threshold

Density evolution would not be such an useful tool if it could not be simulated efficiently. The idea is to estimate numerically the distributions of the density evolution variables $\{h^{(t)}, u^{(t)}\}$. As already discussed this gives access to a number of statistics on BP decoding, such as the bit error rate $\mathrm{P}_{\mathrm{b}}^{(t)}$ after $t$ iterations in the large blocklength limit.

A possible approach consists in representing the distributions by samples of some fixed size. Within statistical physics this is sometimes called the *population dynamics algorithm* (and made its first appearance in the study of the localization transition on Cayley trees [46]). Although there exist more

[11] The suspicious reader will notice that this is not exactly a particular case of the previous statement, because $f(x) = \mathbb{I}(x < 0) + \frac{1}{2}\mathbb{I}(x = 0)$ is not a continuous function.

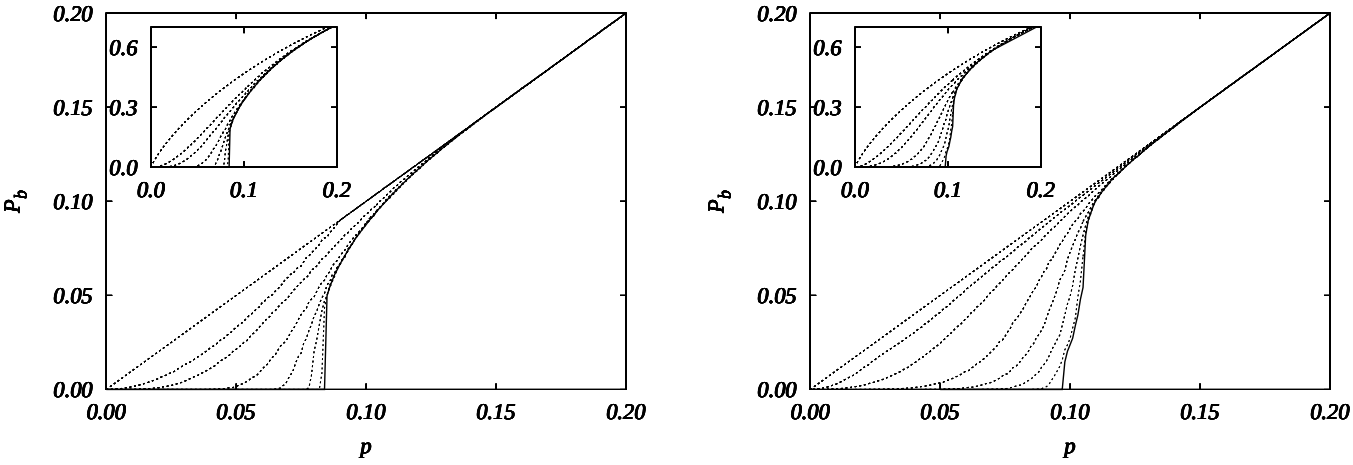


Figure 15: The performances of two LDPC ensembles as predicted by a numerical implementation of density evolution. On the left, the $(3, 6)$ regular ensemble. On the right, an optimized irregular ensemble. Dotted curves refer (from top to bottom) to $t = 0$, 1, 2, 5, 10, 20, 50 iterations, and bold continuous lines to the limit $t \to \infty$. In the inset we plot the expected conditional entropy $\mathbb{E}\, H(X_i|\overline{\nu}_i^{(t)})$.

efficient alternatives in the coding context (mainly based on Fourier transform, see [49, 48]), we shall describe population dynamics because it is easily programmed.

Let us describe the algorithm within the setting of a general message passing decoder, cf. Eq. (4.17). Given an integer $\mathcal{N} \gg 1$, one represent the messages distributions with two samples of size $\mathcal{N}$: $\mathfrak{P}^{(t)} = \{\nu_1^{(t)}, \dots, \nu_{\mathcal{N}}^{(t)}\}$, and $\widehat{\mathfrak{P}}^{(t)} = \{\widehat{\nu}_1^{(t)}, \dots, \widehat{\nu}_{\mathcal{N}}^{(t)}\}$. Such samples are used as proxy for the corresponding distributions. For instance, one would approximate an expectation as

$$\mathbb{E} f(\nu^{(t)}) \approx \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} f(\nu_i^{(t)})\,. \tag{4.22}$$

The populations are updated iteratively. For instance $\mathfrak{P}^{(t+1)}$ is obtained from $\widehat{\mathfrak{P}}^{(t)}$ by generating $\nu_1^{(t+1)}, \dots, \nu_{\mathcal{N}}^{(t+1)}$ independently as follows. For each $i \in [\mathcal{N}]$, draw indices $b_1(i), \dots, b_l(i)$ independently and uniformly at random from $[\mathcal{N}]$, and generate $y_i$ with distribution $Q(y|0)$. Then compute $\nu_i^{(t+1)} = \Phi(\{\widehat{\nu}_{b_n(i)}^{(t)}\}; y_i)$ and store it in $\mathfrak{P}^{(t+1)}$.

An equivalent description consists in saying that we proceed as if $\widehat{\mathfrak{P}}^{(t)}$ exactly represents the distribution of $u^{(t)}$ (which in this case would be discrete). If this was the case, the distribution of $h^{(t+1)}$ would be composed of $|\mathcal{A}| \cdot \mathcal{N}^{l-1}$ Dirac deltas. In order not to overflow memory, the algorithm samples $\mathcal{N}$ values from such a distribution. Empirically, estimates of the form (4.22) obtained through population dynamics have systematic errors of order $\mathcal{N}^{-1}$ and statistical errors of order $\mathcal{N}^{-1/2}$ with respect to the exact value.

In Fig. 15 we report the results of population dynamics simulations for two different LDPC ensembles, with respect to the BSC. We consider two performance measures: the bit error rate $\mathrm{P}_{\mathrm{b}}^{(t)}$ and the bit conditional entropy $H^{(t)}$. The latter is defined as

$$H^{(t)} = \lim_{N \to \infty} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}\, H(X_i|\overline{\nu}_i^{(t)})\,, \tag{4.23}$$

and encodes the uncertainty about bit $x_i$ after $t$ BP iterations. It is intuitively clear that, as the algorithm

| $l$ | $k$ | $R$ | $p_{\rm d}$ | Shannon limit |
|---|---|---|---|---|
| 3 | 4 | 1/4 | 0.1669(2) | 0.2145018 |
| 3 | 5 | 2/5 | 0.1138(2) | 0.1461024 |
| 3 | 6 | 1/2 | 0.0840(2) | 0.1100279 |
| 4 | 6 | 1/3 | 0.1169(2) | 0.1739524 |

Table 1: Belief propagation thresholds for a few regular LDPC ensembles.

progresses, the bit estimates improve and therefore $\mathrm{P}_{\rm b}^{(t)}$ and $H^{(t)}$ should be monotonically decreasing functions of the number of iterations. Further, they are expected to be monotonically increasing functions of the crossover probability $p$. Both statement can be easily checked on the above plots, and can be proved rigorously as well.

Since $\mathrm{P}_{\rm b}^{(t)}$ is non-negative and decreasing in $t$, it has a finite limit

$$\mathrm{P}_{\rm b}^{\rm BP} \equiv \lim_{t\to\infty} \mathrm{P}_{\rm b}^{(t)} \,, \tag{4.24}$$

which is itself non-decreasing in $p$. The limit curve $\mathrm{P}_{\rm b}^{\rm BP}$ is estimated in Fig. 15 by choosing $t$ large enough so that $\mathrm{P}_{\rm b}^{(t)}$ is independent of $t$ within the numerical accuracy.

Since $\mathrm{P}_{\rm b}^{\rm BP} = \mathrm{P}_{\rm b}^{\rm BP}(p)$ is a non-decreasing function of $p$, one can define the *BP threshold*

$$p_{\rm d} \equiv \sup\left\{\, p \in [0,1/2] \,:\, \mathrm{P}_{\rm b}^{\rm BP}(p) = 0 \,\right\} \,. \tag{4.25}$$

Analogous definitions can be provided for other channel families such as the BEC($\epsilon$). In general, the definition (4.25) can be extended to any family of BMS channels BMS($p$) indexed by a real parameter $p \in I$, $I \subseteq \mathbb{R}$ being an interval (obviously the sup will be then taken over $p \in I$). The only condition is that the family is 'ordered by physical degradation'. We shall not describe this concept formally, but limit ourselves to say that that $p$ should be an 'honest' noise parameter, in the sense that the channel worsen as $p$ increases.

Analytical upper and lower bounds can be derived for $p_{\rm d}$. In particular it can be shown that it is strictly larger than 0 (and smaller than 1/2) for all LDPC ensembles with minimum variable degree at least 2. Numerical simulation of density evolution allows to determine it numerically with good accuracy. In Table 4.5 we report the results of a few such results.

Let us stress that the threshold $p_{\rm d}$ has an important practical meaning. For any $p < p_{\rm d}$ one can achieve arbitrarily small bit error rate with high probability by just picking one random code from the ensemble LDPC and using BP decoding and running it for a large enough (but independent of the blocklength) number of iterations. For $p > p_{\rm d}$ the bit error rate is asymptotically lower bounded by $\mathrm{P}_{\rm b}^{\rm BP}(p) > 0$ for any fixed number of iterations. In principle it could be that after, let's say $n^a$, $a > 0$ iterations a lower bit error rate is achieved. However simulations show quite convincingly that this is not the case.

In physics terms the algorithm undergoes a phase transition at $p_{\rm d}$. At first sight, such a phase transition may look entirely dependent on the algorithm definition and not 'universal' in any sense. As we will discuss in the next section, this is not the case. The phase transition at $p_{\rm d}$ is somehow intrinsic to the underlying measure $\mu(\underline{x})$, and has a well studied counterpart in the theory of mean field disordered spin models.

Apart from the particular channel family, the BP threshold depends on the particular code ensemble, i.e. (for the case considered here) on the code ensemble. It constitutes therefore a primary measure of the 'goodness' of such a pair. Given a certain design rate $R$, one would like to make $p_{\rm d}$ as large as possible. This has motivated the introduction of code ensembles that generalize the regular ones studied here (starting from 'irregular' ones). Optimized ensembles have been shown to allow for exceptionally

good performances. In the case of the erasure channel, they allowed to saturate Shannon's fundamental limit [27]. This is an important approach to the design of LDPC ensembles.

Let us finally mention that the BP threshold was defined in Eq. (4.25) in terms of the bit error rate. One may wonder whether a different performance parameter may yield a different threshold. As long as such parameter can be written in the form $\frac{1}{N}\sum_i f(h_i^{(t)})$ this is not the case. More precisely

$$p_{\rm d} = \sup\left\{ p \in I \,:\, h^{(t)} \stackrel{\rm d}{\to} +\infty \right\}\,, \tag{4.26}$$

where, for the sake of generality we assumed the noise parameter to belong to an interval $I \subseteq \mathbb{R}$. In other words, for any $p < p_{\rm d}$ the distribution of BP messages becomes a delta at plus infinity.

## 4.6 Belief Propagation versus MAP Decoding

So far we have seen that detailed predictions can be obtained for the performance of LDPC ensembles under message passing decoding (at least in the large blocklength limit). In particular the threshold noise for reliable communication is determined in terms of a distributional recursion (density evolution). This recursion can in turn be efficiently approximated numerically, leading to accurate predictions for the threshold.

It would be interesting to compare such predictions with the performances under optimal decoding strategies. Throughout this section we shall focus on symbol MAP decoding, which minimizes the bit error rate, and consider a generic channel family $\{\mathrm{BMS}(p)\}$ ordered[12] by the noise parameter $p$.

Given an LDPC ensemble, let $\mathrm{P}_{\rm b}^{(N)}$ be the expected bit error rate when the blocklength is $N$. The *MAP threshold* $p_{\rm c}$ for such an ensemble can be defined as the largest (or, more precisely, the supremum) value of $p$ such that $\lim_{N\to\infty} \mathrm{P}_{\rm b}^{(N)} = 0$. In other words, for any $p < p_{\rm c}$ one can communicate with an arbitrarily small error probability, by using a random code from the ensemble, provided $N$ is large enough.

By the optimality of MAP decoding, $p_{\rm d} \le p_{\rm c}$. In coding theory some techniques have been developed to prove upper and lower bounds on $p_{\rm c}$ [19, 52]. In particular it is easy to find ensembles for which there exist a gap between the two thresholds (namely $p_{\rm d} < p_{\rm c}$ strictly). Consider for instance $(k, l)$ regular ensembles with a fixed ratio $l/k = 1 - R$. It is then possible to show that, as $k, l \to \infty$, the BP threshold goes to 0 while the MAP threshold approaches the Shannon limit.

This situation is somewhat unsatisfactory. The techniques used to estimate $p_{\rm d}$ and $p_{\rm c}$ are completely different. This is puzzling since the two thresholds can be extremely close and even coincide for some ensembles. Furthermore, we know that $p_{\rm d} \le p_{\rm c}$ by a general argument (optimality of MAP decoding), but this inequality is not 'built in' the corresponding derivations. Finally, it would be interesting to have a sharp estimate for $p_{\rm c}$.

It turns out that a sharp characterization of $p_{\rm c}$ can be obtained through statistical mechanics techniques [42, 58, 38]. The statistical mechanics result has been proved to be a rigorous upper bound for general code ensembles, and it is conjectured to be tight [39, 35].

The starting point is to consider the conditional entropy of the channel input $\underline{x}$ given the output $\underline{y}$, $H_N(\underline{X}|\underline{Y})$. As shown in Eq. (2.16) this is given by the expectation of the log partition function appearing in Eq. (4.1) (apart from a trivial additive factor).

Let $\mathsf{f}_N = \mathbb{E}H_N(\underline{X}|\underline{Y})/N$ denote the entropy density averaged over the code ensemble. Intuitively speaking, this quantity allows to estimate the typical number of inputs with non-negligible probability for a given channel output. If $\mathsf{f}_N$ is bounded away from 0 as $N \to \infty$, the typical channel output corresponds to an exponential number of (approximately) equally likely inputs. If on the other hand $\mathsf{f}_N \to 0$, the correct input has to be searched among a sub-exponential number of candidates. This leads us to identify[13] the MAP threshold as the largest noise level such that $\mathsf{f}_N \to 0$ as $N \to \infty$.

[12] Such that the channel worsen as $p$ increases. Examples are the binary symmetric or binary erasure channels.

[13] A rigorous justification of this identification can be obtained using Fano's inequality.

The Bethe free energy provides a natural way to approximate log-partition functions on sparse graphs. It is known to be exact if the underlying graph is a tree and its stationary points are in correspondence with the fixed points of BP. In statistical physics terms, it is the correct variational formulation for the Bethe Peierls approximation. In random systems which are locally tree like, it is normally thought to provide the correct $N \to \infty$ limit unless long range correlations set in. These are in turn described through 'replica symmetry breaking' (see below).

As many mean field approximations, the Bethe approximation can be thought of as a way of writing the free energy as a function of a few correlation functions. More specifically, one considers the single-variable marginals $\{b_i(x_i) : \ i \in \{1,\dots,N\}\}$, and the joint distributions of variables involved in a common check node $\{b_a(\underline{x}_a) : \ a \in \{1,\dots,M\}\}$. In the present case the Bethe free energy reads

$$\begin{aligned} F_{\rm B}(\underline{b}) \;\; &= \;\; -\sum_{i=1}^{N}\sum_{x_i} b_i(x_i)\log Q(y_i|x_i) + \\ &\quad + \sum_{a=1}^{M}\sum_{\underline{x}_a} b_a(\underline{x}_a)\log b_a(\underline{x}_a) - \sum_{i=1}^{N}(|\partial i|-1)\sum_{x_i} b_i(x_i)\log_2 b_i(x_i)\,. \end{aligned} \tag{4.27}$$

The marginals $\{b_i(\,\cdot\,)\}$, $\{b_a(\,\cdot\,)\}$ are regarded as variables. They are constrained to be probability distributions (hence non-negative) and to satisfy the marginalization conditions

$$\sum_{x_j,\, j\in\partial a\setminus i} b_a(\underline{x}_a) = b_i(x_i) \qquad \forall i\in\partial a\,, \qquad\qquad \sum_{x_i} b_i(x_i) = 1 \qquad \forall i\,. \tag{4.28}$$

Further, in order to fulfill the parity-check constraints $b_a(\underline{x}_a)$ must be forced to vanish unless $x_{i_a(1)}\oplus\cdots\oplus x_{i_a(k)} = \mathtt{0}$ (as usual we use the convention $0\log 0 = 0$). Since they do not necessarily coincide with the actual marginals of $\mu(\,\cdot\,)$, the $\{b_a\}$, $\{b_i\}$ are sometimes called *beliefs*.

Approximating the log-partition function $-\log Z(\underline{y})$ requires minimizing the Bethe free energy $F_{\rm B}(\underline{b})$. The constraints can be resolved by introducing Lagrange multipliers, that are in turn expressed in terms of two families of real valued messages $\underline{u}\equiv\{u_{a\to i}\}$, $\underline{h}=\{h_{i\to a}\}$. If we denote by $P_u(x)$ the distribution of a bit $x$ whose log likelihood ratio is $u$ (in other words $P_u(0) = 1/(1+e^{-2u})$, $P_u(1) = e^{-2u}/(1+e^{-2u})$), the resulting beliefs read

$$b_a(\underline{x}_a) = \frac{1}{z_a}\,\mathbb{I}_a(\underline{x})\prod_{j\in\partial a} P_{h_{j\to a}}(x_j)\,, \quad b_i(x_i) = \frac{1}{z_i}\,Q(y_i|x_i)\prod_{a\in\partial i} P_{u_{a\to i}}(x_i)\,, \tag{4.29}$$

where we introduced the shorthand $\mathbb{I}_a(\underline{x})$ to denote the indicator function for the $a$-th parity check being satisfied. Using the marginalization conditions (4.28) as well as the stationarity of the Bethe free energy with respect to variations in the beliefs, one obtains the fixed point BP equations

$$h_{i\to a} = B_i + \sum_{b\in\partial i\setminus a} u_{b\to i}\,, \qquad u_{a\to i} \;\; = \;\; \operatorname{atanh}\left\{\prod_{j\in\partial a\setminus i}\tanh h_{j\to a}\right\}. \tag{4.30}$$

These in turn coincide for with the fixed point conditions for belief propagation, cf. Eqs. (4.11).

The Bethe free energy can be written as a function of the messages by plugging the expressions (4.29) into Eq. (4.27). Using the fixed point equations, we get

$$\begin{aligned} F_{\rm B}(\underline{u},\underline{h}) \;\; &= \;\; \sum_{(ia)\in E}\log\left[\sum_{x_i} P_{u_{a\to i}}(x_i)P_{h_{i\to a}}(x_i)\right] - \\ &\quad - \sum_{i=1}^{N}\log\left[\sum_{x_i} Q(y_i|x_i)\prod_{a\in\partial i} P_{u_{a\to i}}(x_i)\right] - \sum_{a=1}^{M}\log\left[\sum_{\underline{x}_a}\mathbb{I}_a(\underline{x})\prod_{i\in\partial a} P_{h_{i\to a}}(x_i)\right]. \end{aligned} \tag{4.31}$$

We are interested in the expectation of this quantity with respect to the code and channel realization, in the $N \to \infty$ limit. We assume that messages are asymptotically identically distributed $u_{a\to i} \stackrel{\mathrm{d}}{=} u$, $h_{i\to a} \stackrel{\mathrm{d}}{=} h$, and that messages incoming in the same node along distinct edges are asymptotically independent. Under these hypotheses we get the limit

$$\lim_{N\to\infty} \frac{1}{N} \mathbb{E}\, F_{\mathrm{B}}(\underline{u}, \widehat{u}) = -\phi_{u,h} + \sum_y Q(y|0) \log_2 Q(y|0)\,, \tag{4.32}$$

where

$$\begin{aligned} \phi_{u,h} \;\equiv\; & -l\, \mathbb{E}_{u,h} \log_2 \left[ \sum_x P_u(x) P_h(x) \right] + \mathbb{E}_y \mathbb{E}_{\{u_i\}} \log_2 \left[ \sum_x \frac{Q(y|x)}{Q(y,0)} \prod_{i=1}^{l} P_{u_i}(x) \right] - \\ & + \frac{l}{k}\, \mathbb{E}_{\{h_i\}} \log_2 \left[ \sum_{x_1 \dots x_k} \mathbb{I}_a(\underline{x}) \prod_{i=1}^{k} P_{h_i}(x_i) \right]. \end{aligned} \tag{4.33}$$

Notice that the random variables $u$, $h$ are constrained by Eq. (4.30), which must be fulfilled in distributional sense. In other words $u$, $h$ must form a fixed point of the density evolution recursion (4.18). Given this proviso, if the above assumptions are correct and the Bethe free energy is a good approximation for the log partition function one expects the conditional entropy per bit to be $\lim_{N\to\infty} \mathsf{f}_N = \phi_{u,h}$. This guess is supported by the following rigorous result.

**Theorem 4.2.** *If $u$, $h$ are symmetric random variables satisfying the distributional identity $u \stackrel{\mathrm{d}}{=} \operatorname{atanh}\left\{\prod_{i=1}^{k-1} \tanh h_i\right\}$, then*

$$\lim_{N\to\infty} \mathsf{f}_N \geq \phi_{u,h}\,. \tag{4.34}$$

It is natural to conjecture that the correct limit is obtained by optimizing the above lower bound, i.e.

$$\lim_{N\to\infty} \mathsf{f}_N = \sup_{u,h} \phi_{u,h}\,, \tag{4.35}$$

where, once again the sup is taken over the couples of symmetric random variables satisfying $u \stackrel{\mathrm{d}}{=} \operatorname{atanh}\left\{\prod_{i=1}^{k-1} \tanh h_i\right\}$. In fact it is easy to show that, on the fixed point, the distributional equation $h \stackrel{\mathrm{d}}{=} B + \sum_{a=1}^{l-1} u_a$ must be satisfied as well. In other words the couple $u$, $h$ must be a density evolution fixed point.

This conjecture has indeed been proved in the case of communication over the binary erasure channel for a large class of LDPC ensembles (including, for instance, regular ones).

The expression (4.35) is interesting because it bridges the analysis of BP and MAP decoding. For instance, it is immediate to show that it implies $p_{\mathrm{d}} \leq p_{\mathrm{c}}$.

Exercise 4: This exercise aims at proving the last statement.

(*a*) Recall that $u, h = +\infty$ constitute a density evolution fixed point for any noise level. Show that $\phi_{h,u} = 0$ on such a fixed point.

(*b*) Assume that, if any other fixed point exists, then density evolution converges to it (this can indeed be proved in great generality).

(*c*) Deduce that $p_{\mathrm{d}} \leq p_{\mathrm{c}}$.

Evaluating the expression (4.35) implies an a priori infinite dimensional optimization problem. In practice good approximations can be obtained through the following procedure:

1. Initialize $h$, $u$ to a couple of symmetric random variables $h^{(0)}$, $u^{(0)}$.

| $l$ | $k$ | $R$ | $p_{\mathrm{c}}$ | Shannon limit |
|---|---|---|---|---|
| 3 | 4 | 1/4 | 0.2101(1) | 0.2145018 |
| 3 | 5 | 2/5 | 0.1384(1) | 0.1461024 |
| 3 | 6 | 1/2 | 0.1010(2) | 0.1100279 |
| 4 | 6 | 1/3 | 0.1726(1) | 0.1739524 |

Table 2: MAP thresholds for a few regular LDPC ensembles and communication over the BSC($p$).

2. Implement numerically the density evolution recursion (4.18) and iterate it until an approximate fixed point is attained.
3. Evaluate the functional $\phi_{u,h}$ on such a fixed point, after enforcing $u \stackrel{\mathrm{d}}{=} \operatorname{atanh}\left\{\prod_{i=1}^{k-1} \tanh h_i\right\}$ exactly.

The above procedure can be repeated for several different initializations $u^{(0)}$, $h^{(0)}$. The largest of the corresponding values of $\phi_{u,h}$ is then picked as an estimate for $\lim_{N\to\infty} \mathsf{f}_N$.

While his procedure is not guaranteed to exhaust all the possible density evolution fixed points, it allows to compute a sequence of lower bounds to the conditional entropy density. Further, one expects a small finite number of density evolution fixed points. In particular, for regular ensembles and $p > p_{\mathrm{d}}$, a unique (stable) fixed point is expected to exist apart from the no-error one $u, h = +\infty$. In Table 4.6 we present the corresponding MAP thresholds for a few regular ensembles.

For further details on these results, and complete proofs, we refer to [39]. Here we limit ourselves to a brief discussion why the conjecture (4.35) is expected to hold from a statistical physics point of view.

The expression (4.35) corresponds to the 'replica symmetric ansatz' from the present problem. This usually breaks down if some form of long-range correlation ('replica symmetry breaking') arises in the measure $\mu(\,\cdot\,)$. This phenomenon is however not expected to happen in the case at hand. The technical reason is that the so-called Nishimori condition holds for $\mu(\,\cdot\,)$ [38]. This condition generally holds for a large family of problems arising in communications and statistical inference. While Nishimori condition does not provide an easy proof of the conjecture (4.35), it implies a series of structural properties of $\mu(\,\cdot\,)$ that are commonly regarded as incompatible with replica symmetry breaking.

Replica symmetry breaking is instead necessary to describe the structure of 'metastable states' [17]. This can be loosely described as very deep local minima in the energy landscape introduced in Section 4.1. Here 'very deep' means that $\Theta(N)$ bit flips are necessary to lower the energy (number of unsatisfied parity checks) when starting from such minima. As the noise level increases, such local minima become relevant at the so called 'dynamic phase transition'. It turns out that the critical noise for this phase transition coincides with the BP threshold $p_{\mathrm{d}}$. In other words the double phase transition at $p_{\mathrm{d}}$ and $p_{\mathrm{c}}$ is completely analogous to what happens in the mean field theory of structural glasses (see for instance Parisi's lectures at this School). Furthermore, this indicates that $p_{\mathrm{d}}$ has a 'structural' rather than purely algorithmic meaning.

# 5 Belief Propagation Beyond Coding Theory

The success of belief propagation as an iterative decoding procedure has spurred a lot of interest in its application to other statistical inference tasks.

A simple formalization for this family of problems is provided by factor graphs. One is given a factor graph $G = (V, F, E)$ with variable nodes $V$, function nodes $F$, and edges $E$ and considers probability

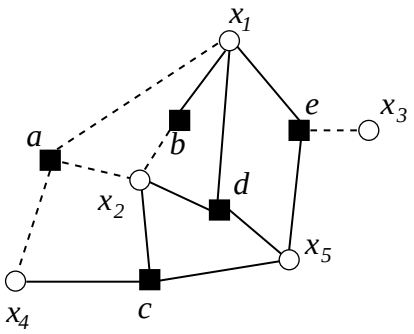


Figure 16: Factor graph representation of the satisfiability formula (5.5). Circle correspond to variables and squares to clauses. Edges are represented as dashed line if the variable is negated in the corresponding clause.

distributions that factorize accordingly

$$\mu(\underline{x}) = \frac{1}{Z} \prod_{a \in F} \psi_a(\underline{x}_{\partial a})\,. \tag{5.1}$$

Here the variables $x_i$ take values in a generic finite alphabet $\mathcal{X}$, and the *compatibility functions* $\psi_a : \mathcal{X}^{\partial a} \to \mathbb{R}_+$ encode dependencies among them. The prototypical problem consists in computing marginals of the distribution $\mu(\,\cdot\,)$, e.g.,

$$\mu_i(x_i) \equiv \sum_{\underline{x}_{\sim i}} \mu(\underline{x})\,. \tag{5.2}$$

Belief propagation can be used to accomplish this task in a fast and distributed (but not necessarily accurate) fashion. The general update rules read

$$\nu_{i\to a}^{(t+1)}(x_i) \propto \prod_{b \in \partial i \setminus a} \widehat{\nu}_{b\to i}^{(t)}(x_i)\,, \qquad \widehat{\nu}_{a\to i}^{(t)}(x_i) \propto \sum_{\{x_j\}} \psi_a(\underline{x}_{\partial a}) \prod_{j \in \partial a \setminus i} \nu_{j\to a}^{(t)}(x_j)\,. \tag{5.3}$$

Messages are then used to estimate local marginals as follows

$$\overline{\nu}_i^{(t+1)}(x_i) \quad \propto \quad \prod_{b \in \partial i} \widehat{\nu}_{b\to i}^{(t)}(x_i)\,. \tag{5.4}$$

The basic theoretical question is of course to establish a relation, if any between $\mu_i(\,\cdot\,)$ and $\overline{\nu}_i(\,\cdot\,)$.

As an example, we shall consider *satisfiability* [23]. Given $N$ Boolean variables $x_i$, $i \in \{1, \dots, N\}$, $x_i \in \{\text{True, False}\}$, a formula is the logical expression obtained by taking the AND of $M$ *clauses*. Each clause is the logical OR of a subset of the variables or their negations. As an example, consider the formula (here $\overline{x}_i$ denotes the negation of $x_i$)

$$(\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_4) \wedge (x_1 \vee \overline{x}_2) \wedge (x_2 \vee x_4 \vee x_5) \wedge (x_1 \vee x_2 \vee \overline{x}_5) \wedge (x_1 \vee \overline{x}_3 \vee x_5)\,. \tag{5.5}$$

An assignment of the $N$ variables satisfies the formula if, for each of the clause, at least one of the involved *literals* (i.e. either the variables or their negations) evaluates to True.

A satisfiability formula admits a natural factor graph representation where each clause is associated to a factor node and each variable to a variable node. An example is shown in Fig. 16. Admitting that the formula has at least one satisfying assignment, it is natural to associate to a $\mathcal{F}$ the uniform measure

over such assignments $\mu_{\mathcal{F}}(\underline{x})$. It is easy to realize that such a distribution takes the form (5.1) where the compatibility function $\psi_a(\underline{x}_{\partial a})$ takes value 1 if the assignment $\underline{x}$ satisfies clause $a$ and 0 otherwise.

Satisfiability, i.e., the problem of finding a solution to a satisfiability formula or proving that it is usatisfiable, is one of the prototypical NP-complete problems. Given a satisfiability formula, computing marginals with respect to the associated distribution $\mu_{\mathcal{F}}(\,\cdot\,)$ is relevant for tasks such as counting the number of solutions of $\mathcal{F}$ or sampling them uniformly. These are well known *#-P complete* problems.[14]

The currently best algorithms for solving random instances for the $K$-SAT problem are based on a variant of BP, which is called *survey propagation* [11, 33, 11].

## 5.1 Proving Correctness through Correlation Decay

A simple trick to bound the error incurred by BP consists in using the correlation decay properties [1, 57] of the measure $\mu(\,\cdot\,)$. Let $i \in \{1, \dots, N\}$ be a variable index and denote by $\underline{x}_t$ the set of variables sitting on nodes at distance $t$ from $i$. Further, denote by $\underline{x}_{\geq t}$ the set of variables whose distance from $i$ is *at least* $t$. Then the local structure of the probability distribution (5.1)

$$\mu_i(x_i) = \sum_{\underline{x}_{\geq t}} \mu(x_i|\underline{x}_{\geq t})\mu(\underline{x}_{\geq t}) = \sum_{\underline{x}_t} \mu(x_i|\underline{x}_t)\mu(\underline{x}_t)\,. \tag{5.6}$$

Let $\mathsf{B}_i(t)$ denote the subgraph induced by nodes whose distance from $i$ is *at most* $t$, and $\partial\mathsf{B}_i(t)$ its boundary (nodes whose distance from $i$ is exactly $t$). Further, for any $j \in \partial\mathsf{B}_i(t)$ let $a(j)$ be the unique function nodes inside $\mathsf{B}_i(t)$ that is adjacent to $j$. It is intuitively clear that belief propagation computes the marginal at $i$ *as if* the graph did not extend beyond $\mathsf{B}_i(t)$. More precisely, if the initial condition $\nu^{(0)}_{i\to a}(x_i)$ is properly normalized, then we have the exact expression

$$\overline{\nu}^{(t)}_i(x_i) = \sum_{\underline{x}_t} \mu(x_i|\underline{x}_t) \prod_{j\in\partial\mathsf{B}_i(t)} \nu^{(0)}_{j\to a(j)}(x_j)\,. \tag{5.7}$$

As a consequence of Eq. (5.6) and (5.7) we have

$$|\mu_i(x_i) - \overline{\nu}^{(t)}_i(x_i)| \leq \sup_{\underline{x}_t,\underline{x}'_t} |\mu(x_i|\underline{x}_t) - \mu(x_i|\underline{x}'_t)|\,. \tag{5.8}$$

This provides an upper bound on the error incurred by BP when computing the marginal of $x_i$ base on the local structure of the underlying graph in terms of the influence of far away variables. To make things fully explicit, assume that the graph has *girth*[15] $g$ and that $\sup_{\underline{x}_t,\underline{x}'_t} |\mu(x_i|\underline{x}_t) - \mu(x_i|\underline{x}'_t)| \leq \exp(-\kappa t)$ for some positive $\kappa$. This implies

$$|\mu_i(x_i) - \overline{\nu}^{(t)}_i(x_i)| \leq e^{-\kappa g/2}\,. \tag{5.9}$$

As an example of such error estimates, we shall consider *random $k$-satisfiability* [16]. This is a standard model to generate 'synthetic' satisfiability formulae. It amounts to picking a formula uniformly at random among all the ones including $N$ variables and $M = N\alpha$ $k$-clauses (a $k$-clause is a clause that involve *exactly* $k$ distinct variables). We shall of course limit to $k \geq 2$, the case $k = 1$ being trivial.

Consider a uniformly random variable node in the factor graph associated to a random formula, and its depth-$t$ neighborhood $\mathsf{B}_i(t)$. Proceeding as in the previous section it is not hard to show that, for any fixed $t$, $\mathsf{B}_i(t)$ is with high probability (as $N \to \infty$) a tree. An appropriate model the distribution of such a tree, is given by the tree ensemble $\mathsf{T}_*(t)$ described as follows. For $t = 0$, it is the graph containing a unique variable node. For any $t \geq 1$, start by a single variable node (the root) and add $l \stackrel{\mathrm{d}}{=} \mathsf{Poisson}(k\alpha)$ clauses, each one including the root, and $k-1$ new variables (first generation variables).

[14] The notation #-P refers to the hardness classification for counting problems.

[15] Recall that the girth of a graph is the length of its shortest cycle.

For each one of the $l$ clauses, the corresponding literals are non-negated or negated independently with equal probability. If $t \geq 2$, generate an independent copy of $\mathsf{T}_*(t-1)$ for each variable node in the first generation and attach it to them.

Assume that, for a typical random tree formula $\mathsf{T}(t)$, the marginal distribution of the variable at the root is weakly dependent on the values assigned at the boundary. Following the above lines, one can use this fact to prove that BP computes good approximations for the marginals in a random $k$-SAT formula. In fact it turns out that an estimate of the form[16]

$$\mathbb{E}_{\mathsf{T}(t)} \sup_{\underline{x}_t, \underline{x}'_t} |\mu(x_i|\underline{x}_t) - \mu(x_i|\underline{x}'_t)| \leq e^{-\kappa t} \tag{5.10}$$

can be proved if the clause density $\alpha$ stays below a threshold $\alpha_{\rm u}(k)$ that is estimated to behave as $\alpha_{\rm u}(k) = \frac{2\log k}{k}[1 + o_k(1)]$.

While we refer to the original paper [40] for the details of the proof we limit ourselves to noticing that the left hand side of Eq. (5.10) can be estimated efficiently using a density evolution procedure. This allows to estimate the threshold $\alpha_{\rm u}(k)$ numerically. Consider in fact the log-likelihood (here we are identifying $\{\text{True}, \text{False}\}$ with $\{+1, -1\}$)

$$h^{(t)}(\underline{x}_t) = \frac{1}{2} \log \frac{\mu(+1|\underline{x}_t)}{\mu(-1|\underline{x}_t)} \,. \tag{5.11}$$

This quantity depends on the assignment of the variables on the boundary, $\underline{x}_t$. Since we are interested on a *uniform* bound over the boundary, let us consider the extreme cases

$$\overline{h}^{(t)} = \max_{\underline{x}_t} h^{(t)}(\underline{x}_t) \,, \qquad \underline{h}^{(t)} = \min_{\underline{x}_t} h^{(t)}(\underline{x}_t) \,. \tag{5.12}$$

It is then possible to show that the couple $(\overline{h}^{(t)}, \underline{h}^{(t)})$ obeys a recursive distributional equation that, as mentioned, can be efficiently implemented numerically.

# 6 Belief Propagation Beyond the Binary Symmetric Channel

So far we have considered mainly the case of transmission over BMS channels, our reference example being the BSC. There are many other channel models that are important and are encountered in practical situations. Fortunately, it is relatively straightforward to extend the previous techniques and statements to a much larger class, and we review a few such instances in this section.

## 6.1 Binary Memoryless Symmetric Channels

In order to keep the notation simple, we assumed channel output to belong to a finite alphabet $\mathcal{A}$. In our main example, the BSC, we had $\mathcal{A} = \{0, 1\}$. But in fact all results are valid for a wider class of *binary memoryless symmetric (BMS) channels*. One can prove that there is no loss of generality in assuming the output alphabet to be the real line $\mathbb{R}$ (eventually completed with $\overline{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$).

Let $\underline{y} = (y_1, \dots, y_N)$ be the vector of channel outputs on input $\underline{x} = (x_1, \dots, x_N)$. For a BMS the input is binary, i.e. $\underline{x} \in \{0, 1\}^N$. Further the channel is *memoryless*, i.e. the probability density of getting $\underline{y} \in \overline{\mathbb{R}}^N$ at output when the input is $\underline{x}$, is

$$Q(\underline{y}|\underline{x}) = \prod_{t=1}^{N} Q(y_t|x_t).$$

[16] Here the sup is taken over assignments $\underline{x}_t$, $\underline{x}'_t$ that can be extended to solutions of $\mathsf{T}(t)$.

Finally, the *symmetry* property can be written without loss of generality, as $Q(y_t|x_t = \mathtt{1}) = Q(-y_t|x_t = \mathtt{0})$.

One of the most important elements in this class is the *additive white Gaussian noise* (AWGN) channel, defined by

$$y_t = x_t + z_t\,, \qquad t \in \{1, \ldots, N\}\,,$$

where the sequence $\{z_t\}$ is i.i.d. consisting of Gaussian random variables with mean zero and variance $\sigma^2$. It is common in this setting to let $x_i$ take values in $\{+1, -1\}$ instead of $\{\mathtt{0}, \mathtt{1}\}$ as we have assumed so far. The AWGNC transition probability density function is therefore

$$Q(y_t|x_t) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(y-x)^2}{2\sigma^2}}\,.$$

The AWGNC is the basic model of transmission of an electrical signal over a cable (here the noise is due to thermal noise in the receiver) and it is also a good model of a wireless channel in free space (e.g., transmission from a satellite).

Although the class of BMS channels is already fairly large, it is important in practice to go beyond it. The extension to the non-binary case is quite straightforward and so we will not discuss it in detail. The extension to channels with memory or the asymmetric case are more interesting and so we present them in the subsequent two sections.

## 6.2 Channels With Memory

Loosely speaking, in a memoryless channel the channel acts on each transmitted bit independently. In a channel with memory, on the other hand, the channel acts generally on the whole block of input bits together. An important special case of a channel with memory is if the channel can be modeled as a Markov chain, taking on a sequence of "channel states." Many physical channels posses this property and under this condition the message-passing approach can still be applied. For channels with memory there are two problems. First, we need to determine the capacity of the channel. Second, we need to devise efficient coding schemes that achieve rates close to this capacity. It turns out that both problems can be addressed in a fairly similar framework. Rather than discussing the general case we will look at a simple but typical example.

Let us start by computing the information rate/capacity of channels with memory, assuming that the channel has a Markov structure. As we discussed in Section 2.5 in the setting of BMS channels, the channel capacity can be expressed as the difference of two entropies, namely as $H(X) - H(X|Y)$. Here, $X$ denotes the binary input and $Y$ denotes the observation at the output of the channel whose input is $X$. Given two random variables $X$, $Y$, this entropy difference is called the *mutual information* and is typically denoted by $I(X;Y) = H(X) - H(X;Y)$.

A general channel, is defined by a channel transition probability $Q(y_1^N|x_1^N)$ (here and below $x_1^N$ denotes the vector $(x_1, \ldots, x_N)$). In order to define a joint distribution of the input and output vectors, we have to prescribe a distribution on the channel input, call it $p(x_1^N)$. The channel capacity is obtained by maximizing the mutual information over all possible distributions of the input, and eventually taking the $N \to \infty$ limit. In formulae

$$\mathtt{C}(Q) = \lim_{N\to\infty} \sup_{p(\,\cdot\,)} I(X_1^N; Y_1^N)/N\,.$$

For BMS channels it is possible to show that the maximum occurs for the uniform prior: $p(x_1^N) = 1/2^N$. Under this distribution, $I(X_1^N; Y_1^N)/N$ is easily seen not to depend on $N$ and we recover the expression in Sec. 2.5.

For channels with memory we have to maximize the mutual information over all possible distributions over $\{\mathtt{0}, \mathtt{1}\}^N$ (a space whose dimension is exponential in $N$), and take the limit $N \to \infty$. An easier task is

to choose a convenient input distribution $p(\,\cdot\,)$ and then compute the corresponding mutual information in the $N \to \infty$ limit:

$$I = \lim_{N\to\infty} I(X_1^N; Y_1^N)/N\,. \tag{6.1}$$

Remarkably, this quantity has an important operational meaning. It is the largest rate at which we can transmit reliably across the channel using a coding scheme such that the resulting input distribution matches $p(\,\cdot\,)$.

To be definite, assume that the channel is defined by a state sequence $\{\sigma_t\}_{t\geq 0}$, taking values in a finite alphabet, such that the joint probability distribution factors in the form

$$p(x_1^n, y_1^n, \sigma_0^n) = p(\sigma_0) \prod_{i=1}^{n} p(x_i, y_i, \sigma_i \mid \sigma_{i-1}). \tag{6.2}$$

We will further assume that the transmitted bits $(x_1, \cdot, x_N)$ are iid uniform in $\{0, 1\}$. The factor graph corresponding to (6.2) is shown in Fig. 17. It is drawn in a somewhat different way compared to the factor graphs we have seen so far. Note that in the standard factor graph corresponding to this factorization all variable nodes have degree two. In such a case it is convenient not to draw the factor graph as a bipartite graph but as a standard graph in which the nodes correspond to the factor nodes and the edges correspond to the variable nodes (which have degree two and therefore connect exactly two factors). Such a graphical representation is also known as *normal* graph or as Forney-style factor graph (FSFG), in honor of Dave Forney who introduced them [15]. Let us now look at a concrete example.

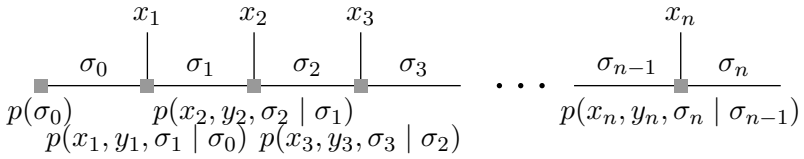


Figure 17: The FSFG corresponding to (6.2).

Example 5:[Gilbert-Elliott Channel] The Gilbert-Elliot channel is a model for a *fading* channel, i.e., a channel where the quality of the channel is varying over time. In this model we assume that the channel quality is evolving according to a Markov chain. In the simplest case there are exactly two states, and this is the original Gilbert-Elliott channel (GEC) model. More precisely, consider the two-state Markov chain depicted in Fig. 18. Assume that $\{X_t\}_{t\geq 1}$ is i.i.d., taking values in $\{\pm 1\}$ with uniform probability.

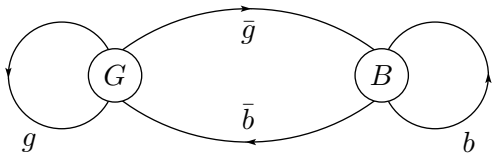


Figure 18: The Gilbert-Elliott channel with two states.

The channel is either in a *good* state, denote it by $G$, or in a *bad* state, call it $B$. In either state the channel is a BSC. Let the crossover probability in the good state be $\epsilon_{\mathrm{G}}$ and in the bad state

be $\epsilon_{\mathrm{B}}$, with $0 \leq \epsilon_{\mathrm{G}} < \epsilon_{\mathrm{B}} \leq 1/2$. Let $P$ be the $2 \times 2$ matrix

$$P = \begin{pmatrix} g & \bar{b} \\ \bar{g} & b \end{pmatrix}$$

which encodes the transition probabilities between the states (the columns indicate the present state and the rows the next state). Define the *steady state* probability vector $p = (p_{\mathrm{G}}, p_{\mathrm{B}})$, i.e., the vector which fulfills $Pp^T = p^T$. This means that in steady state the system spends a fraction $p_{\mathrm{G}}$ of the time in state $G$ and a fraction $p_{\mathrm{B}}$ of the time in state $B$. If we consider e.g. the state $G$ then the detailed balance condition reads $p_{\mathrm{G}}\bar{g} = p_{\mathrm{B}}\bar{b}$. From this we get $p = (\bar{b}/(\bar{g}+\bar{b}), \bar{g}/(\bar{g}+\bar{b}))$. More generally, let us assume that we have $s$ states, $s \in \mathbb{N}$, and that the channel in state $i$, $i \in [s]$, is the BSC($\epsilon_i$). Let $P$ be the $s \times s$ matrix encoding the transition probabilities between these states. Let $p$ denote the steady-state probability distribution vector. If $(I - P^T + E)$ is invertible then a direct check shows that $p = e(I - P^T + E)^{-1}$, where $e$ is the all-one vector of length $s$, $I$ is the $s \times s$ identity matrix and $E$ is the $s \times s$ all-one matrix.

Note that the state sequence is ergodic as long as the Markov chain is irreducible (i.e. there is a path of strictly positive probability from any state to any other state) and aperiodic (i.e. there there exists such a path for any number of steps large enough). In the original Gilbert-Elliot model this is true as long as $0 < g, b < 1$.

Consider the computation of the maximal rate at which we can transmit reliably. We have

$$I(X_1^N; Y_1^N) = H(Y_1^N) - H(Y_1^N | X_1^N).$$

Let us see how we can compute $\lim_{N\to\infty} H(Y_1^N)/N$. Because of the ergodicity assumption on the state sequence, $-\frac{1}{N}\log p(y_1^N)$ converges with probability one to $\lim_{N\to\infty} H(Y_1^N)/N$. It follows that if we can compute $-\frac{1}{N}\log p(y_1^N)$ for a very large sequence, then with high probability the value will be close to the desired entropy rate. Instead of computing $p(y_1^N)$, let us compute $p(\sigma_N, y_1^N)$. From this we trivially get our desired quantity by summing,

$$p(y_1^N) = \sum_{\sigma_N} p(\sigma_N, y_1^N).$$

Note that

$$\begin{aligned} p(\sigma_N, y_1^N) &= \sum_{x_N, \sigma_{N-1}} p(x_N, \sigma_{N-1}, \sigma_N, y_1^N) \\ &= \sum_{x_N, \sigma_{N-1}} \underbrace{p(x_N, \sigma_N, y_N \mid \sigma_{N-1})}_{\text{kernel}} \underbrace{p(\sigma_{N-1}, y_1^{N-1})}_{\text{message}}. \end{aligned} \tag{6.3}$$

From this we see that $p(\sigma_N, y_1^N)$ can be computed recursively. In fact this recursion corresponds to running the BP message-passing rules on the factor graph depicted in Fig. 17 (which is a tree): denote the message which is passed along the edge labeled by $\sigma_N$ by $\nu_N(\sigma_N)$. Then according to the BP message-passing rules we have

$$\nu_N(\sigma_N) = \sum_{x_N, \sigma_{N-1}} p(x_N, \sigma_N, y_N \mid \sigma_{N-1})\, \nu_{N-1}(\sigma_{N-1}).$$

If we compare this to the recursion stated in (6.3) we see that these two recursions are identical. In other words, $\nu_N(\sigma_N) = p(\sigma_N, y_1^N)$, so that

$$\lim_{N\to\infty} H(Y_1^N)/N = -\lim_{N\to\infty} \log\Big(\sum_{\sigma_N} \nu_N(\sigma_N)\Big)/N. \tag{6.4}$$

From a practical perspective it is typically more convenient to pass *normalized* messages $\tilde{\nu}_N(\sigma_N)$ so that $\sum_\sigma \tilde{\nu}_N(\sigma_N) = 1$. The first message $\nu_0(\sigma_0) = p(\sigma_0)$ is already a probability distribution and, hence, normalized, $\tilde{\nu}_0(\sigma_0) = \nu_0(\sigma_0)$. Compute $\nu_1(\sigma_1)$ and let $\lambda_1 = \sum_{\sigma_1} \nu_1(\sigma_1)$. Define $\tilde{\nu}_1(\sigma_1) = \nu_1(\sigma_1)/\lambda_1$. Now note that by definition of the message-passing rules all subsequent messages in the case of rescaling differ from the messages which are sent in the unscaled case only by this scale factor. Therefore, if $\lambda_i$ denotes the normalization constant by which we have to divide at step $i$ so as to normalize the message then $\tilde{\nu}_N(\sigma_N) = \nu_N(\sigma_N)/(\prod_{i=1}^N \lambda_i)$. It follows that

$$\begin{aligned}\lim_{N\to\infty} H(Y_1^N)/N &= -\lim_{N\to\infty} \log\Big(\sum_{\sigma_N} \alpha_N(\sigma_N)\Big)/N \\ &= -\lim_{N\to\infty} \log\Big((\prod_{i=1}^N \lambda_i)\sum_{\sigma_N} \tilde{\alpha}_N(\sigma_N)\Big)/N = \lim_{N\to\infty}\Big(\sum_{i=1}^N \log(\lambda_i)\Big)/N.\end{aligned}$$

It remains to compute $H(Y_1^N \mid X_1^N)$. We write $H(Y_1^N \mid X_1^N)/N = H(Y_1^N, X_1^N)/N - H(X_1^N)/N$. The second part is trivial since the inputs are i.i.d. by assumption so that $H(X_1^N)/N = 1$. For the term $H(Y_1^N, X_1^N)/N$ we use the same technique as for the computation of $H(Y_1^N)/N$. Because of the ergodicity assumption on the state sequence, $-\frac{1}{N}\log p(y_1^N, x_1^N)$ converges with probability one to $\lim_{N\to\infty} H(Y_1^N, X_1^N)/N$. We write $p(y_1^N, x_1^N) = \sum_{\sigma_N} p(\sigma_N, y_1^N, x_1^N)$ and use the factorization

$$\begin{aligned}p(\sigma_N, y_1^N, x_1^N) &= \sum_{\sigma_{N-1}} p(\sigma_{N-1}, \sigma_N, y_1^N, x_1^N) \\ &= \sum_{\sigma_{N-1}} \underbrace{p(x_N, \sigma_N, y_N \mid \sigma_{N-1})}_{\text{kernel}} \cdot \quad \underbrace{p(\sigma_{N-1}, y_1^{N-1}, x_1^{N-1})}_{\text{message}}.\end{aligned}$$

In words, we generate a random instance $X_1^N$ and $Y_1^N$ and run the BP algorithm on the FSFG shown in Fig. 17 assuming that *both* $Y_1^N$ and $X_1^N$ are 'quenched.' Taking the logarithm, multiplying by minus one and normalizing by $1/N$ gives us an estimate of the desired entropy.

Now that we can compute the maximal rate at which we can transmit reliably, let us consider coding. The symbol MAP decoder is

$$\begin{aligned}\widehat{x}_i(\underline{y}) &= \text{argmax}_{x_i} p(x_i \mid y_1^N) \\ &= \text{argmax}_{x_i} \sum_{\{x_j,\, j\neq i\}} p(x_1^N, y_1^N, \sigma_0^N) \\ &= \text{argmax}_{x_i} \sum_{\{x_j,\, j\neq i\}} p(\sigma_0)\prod_{j=1}^N p(x_j, y_j, \sigma_j \mid \sigma_{j-1})\mathbb{I}_{\mathfrak{C}}(x_1^N).\end{aligned}$$

In words, the FSFG in Fig. 17 describes also the factorization for the message-passing decoder if we add to it the factor nodes describing the definition of the code. As always, this factor graph together with the initial messages stemming from the channel completely specify the message-passing rules, except for the message-passing schedule. Let us agree that we alternate one round of decoding with one round of channel estimation. No claim as to the optimality of this scheduling rule is made.

Notice that the correlations induced by the markovian structure of the channel are in general short ranged in time. This is analogous to what happens with a one-dimensional spin model, whose correlation length is always finite (at non-zero temperature). A good approximation to the above message passing schedule is therefore obtained by a 'windowed' decodes. This means that the state at time $t$ is estimated only of the basis of observations between time $t - R$ and $t + R$, for some finite $R$.

Assuming windowed decoding for channel estimation, it is not hard to show that after a fixed number of iterations, the decoding neighborhood is again asymptotically tree-like. In the case of the GEC the channel symmetry can be used to reduce to the all-zero codeword. Therefore, we can employ the technique of density evolution to determine thresholds and to optimize the ensembles.

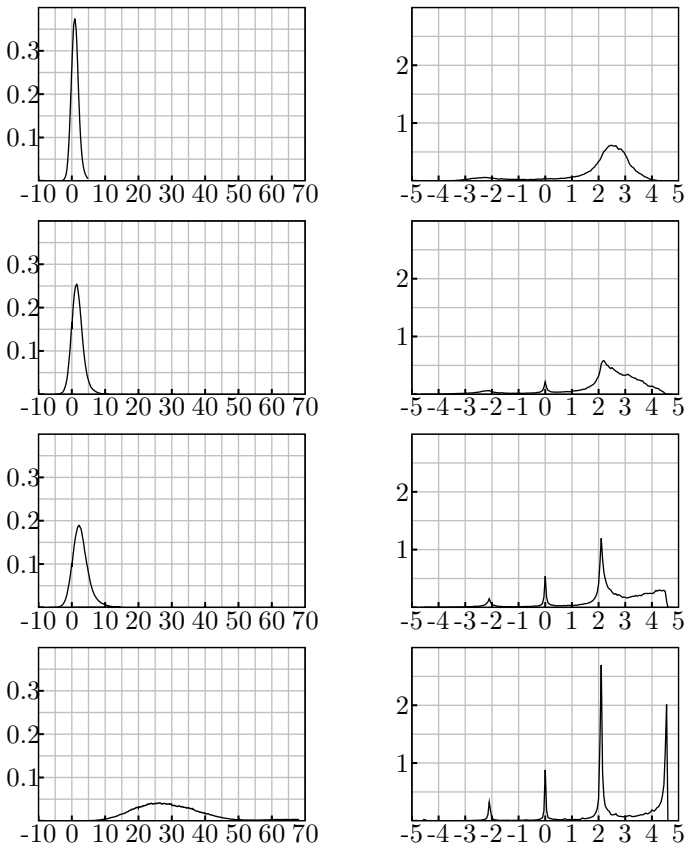


Figure 19: Density evolution for the GEC at iteration 1, 2, 4, and 10. The left pictures show the densities of the messages which are passed from the code towards the part of the FSFG which estimates the channel state. The right hand side shows the density of the messages which are the estimates of the channel state and which are passed to the FSFG corresponding to the code.

Example 6:[GEC: State Estimation] For the case of transmission over the GEC the iterative decoder implicitly also estimates the state of the channel. Let us demonstrate this by means of the following example. We pick a GEC with three states. Let

$$P = \begin{pmatrix} 0.99 & 0.005 & 0.02 \\ 0.005 & 0.99 & 0.02 \\ 0.005 & 0.005 & 0.96 \end{pmatrix},$$

which has a steady state probability vector of $p \approx (0.4444, 0.4444, 0.1112)$. Finally, let the channel parameters of the BSC in these three states be $(\epsilon_1, \epsilon_2, \epsilon_3) \approx (0.01, 0.11, 0.5)$. This corresponds to an *average* error probability of $\epsilon_{\text{avg}} = \sum_{i=1}^{3} p_i \epsilon_i \approx 0.108889$. Using the methods described above, the capacity of this channel (assuming uniform inputs) can be computed to be equal to $C \approx 0.583$ bits per channel use. This is markedly higher than $1 - h(\epsilon_{\text{avg}}) \approx 0.503444$, which is the capacity of the BSC($\epsilon_{\text{avg}}$). The last channel is the channel which we experience if we ignore the Markov structure.

Fig. 19 shows the evolution of the densities for an optimized ensemble of rate $r \approx 0.5498$. The pictures on the right correspond to the messages which are passed from the part of the factor

graph which estimates the state towards the part of the factor graph which describes the code. These messages therefore can be interpreted as the current estimate of the state the channel is in at a given point in time. Note that after 10 iterations 5 clear peaks emerge. These peaks are at $\pm\log(0.99/0.01) \approx \pm 4.595$, $\pm\log(0.9/0.1) \approx \pm 2.197$, $\pm\log(0.5/0.5) = 0$. They correspond to the received likelihoods in the three possible channel states. In other words, the emergence of the peaks shows that at this stage the system has identified the channel states with high reliability. This is quite pleasing. Although the channel state is not known to the receiver and can not be observed directly, in the region where the iterative decoder works reliably it also automatically estimates the channel state with high confidence.

Although we only looked a very particular example it is quite typical of the general situation: as long as the channel memory can be described by a Markov chain the factor graph approach applies and we can use message-passing schemes to construct efficient coding schemes [14, 4, 21, 22, 25, 43].

## 6.3 Asymmetric Channels - The Z Channel

Let us now consider the second generalization, namely the case of *non-symmetric* channels.

Consider the channel depicted on the right of Fig. 1. For obvious reasons it is called the Z channel (ZC). This channel has binary input and it is memoryless but it is *not* symmetric. Nevertheless, essentially the same type of analysis which we performed in Section 4 can be applied to this case as well. Symmetry is therefore a *nice* property to have but it is *not essential*.

Consider the capacity of this channel. Since the channel is not symmetric the capacity is not necessarily given by the mutual information between channel input and channel output for a uniform input distribution of the input. We must instead maximize the mutual information over the input distribution. Since the channel is memoryless, it can be assumed that the input is given by a sequence of i.i.d. bernoulli variables. Assuming that $p(x_i = \mathtt{0}) = \alpha$, the output distribution is

$$(p(y_i = \mathtt{0}), p(y_i = \mathtt{1})) = (\alpha\bar{p}, 1 - \alpha\bar{p}),$$

so that the mutual information $I_\alpha(X;Y)$ for a fixed $\alpha$ is equal to

$$I_\alpha(X;Y) = H(Y) - H(Y \mid X) = h(\alpha\bar{p}) - \alpha h(p). \tag{6.5}$$

Some calculus reveals that the optimal choice of $\alpha$ is

$$\alpha(p) = \frac{p^{p/\bar{p}}}{1 + \bar{p}p^{p/\bar{p}}}, \tag{6.6}$$

so that

$$C_{\mathrm{ZC}(p)} = h(\alpha(p)\bar{p}) - \alpha(p)h(p).$$

Fig. 20 compares $C_{\mathrm{ZC}(p)}$ with $I_{\alpha=\frac{1}{2}}(X;Y)$, i.e., it compares the capacity with the transmission rate which is achievable with *uniform* input distribution. This is important and surprising – only little is lost by insisting on an uniform input distribution: the rate which is achievable by using a uniform input distribution is at least a fraction $\frac{1}{2}e\ln(2) \approx 0.924$ of capacity over the entire range of $p$ (with equality when $p$ approaches one). Even more fortunate, from this perspective the Z channel is the extremal case [32, 54]: the information rate of any binary-input memoryless channel when the input distribution is the uniform one is at least a fraction $\frac{1}{2}e\ln(2)$ of its capacity. From the above discussion we conclude that, when dealing with asymmetric channels, not much is lost if we use a binary linear coding scheme (inducing a uniform input distribution).

Consider the density evolution analysis. Because of the lack of symmetry we can no longer make the all-one codeword assumption. Therefore, it seems at first that we have to analyze the behavior of the decoder with respect to each codeword. Fortunately this is not necessary. First note that, since we consider an ensemble average, only the *type* of the codeword matters. More precisely, let us say that a codeword has type $\tau$ if the fraction of zeros is $\tau$. For $\underline{x} \in \mathfrak{C}$, let $\tau(\underline{x})$ be its type. Let us assume that we


Figure 20: Comparison of $C_{\mathrm{ZC}(p)}$ (solid curve) with $I_{\alpha=\frac{1}{2}}(X;Y)$ (dashed curve), both measured in bits.

use an LDPC ensemble whose dominant type is one-half. This means that "most" codewords contain roughly as many zeros as one. Although it is possible to construct degree-distributions which violate this constraint, "most" degree distributions do fulfill it. Under this assumption there exists some strictly positive constant $\gamma$ such that

$$\mathbb{P}\left\{\tau(\underline{x}) \notin [1/2 - \delta/\sqrt{n}, 1/2 + \delta/\sqrt{n}]\right\} \leq e^{-\delta^2\gamma}, \tag{6.7}$$

where the probability is with respect to a uniformly random codeword $\underline{x}$. We can therefore analyze the performance of such a system in the following way: determine the error probability assuming that the type of the transmitted codeword is "close" to the typical one. Since sublinear changes in the type do not figure in the density analysis, this task can be accomplished by a straightforward density evolution analysis. Now add to this the probability that the type of a random codeword deviates significantly from the typical one. The second term can be made arbitrarily small (see right hand side of (6.7)) by choosing $\delta$ sufficiently large.

We summarize: if we encounter a non-symmetric channel and we are willing to sacrifice a small fraction of capacity then we can still use standard LDPC ensembles (which impose a uniform input distribution) to transmit at low complexity. If it is crucial that we approach capacity even closer, a more sophisticated approach is required. We can combine LDPC ensembles with non-linear mappers which map the uniform input distribution imposed by linear codes into a non-uniform input distribution at the channel input in order to bring the mutual information closer to capacity. For a detailed discussion on coding for the $Z$-channel we refer the reader to [34, 59, 5].

# 7 Open Problems

Let us close by reviewing some of the most important open challenges in the channel coding problem.

## 7.1 Order of Limits

Density evolution computes the limit

$$\lim_{t\to\infty} \lim_{N\to\infty} \mathbb{E}[\mathrm{P}_{\mathrm{b}}^{(N,t)}].$$

In words we determined the limiting performance of an ensemble under a *fixed* number of iterations as the blocklength tends to infinity and then let the number of iterations tend to infinity. As we have seen, this limit is relatively easy to compute. What happens if the order of limits is exchanged, i.e., how does the limit

$$\lim_{N\to\infty} \lim_{t\to\infty} \mathbb{E}[\mathrm{P}_{\mathrm{b}}^{(N,t)}]$$

behave? This limit is closer in spirit to the typical operation in practice: for each fixed length the BP decoder continues until no further progress is achieved. We are interested in the limiting performance as the blocklength tends to infinity.

For the BEC it is known that the two limits coincide. If we combine this with the fact that for the BEC the performance is a monotone function in the number of iterations (any further iteration can only make the result better) then we get the important observation that regardless of how we take the limit (jointly or sequentially), as long as both the blocklength as well as the number of iterations tend to infinity we get the same result. From a practical perspective this is comforting to know: it shows that we can expect a certain robustness of the performance with respect to the details of the operation.

It is conjectured that the same statement holds for general BMS channels. Unfortunately, no proof is known.

## 7.2 Finite-Length Performance

The threshold gives an indication of the asymptotic performance: for channel parameters which are better than the threshold sufficiently long codes allow transmission at an arbitrarily low probability of bit error. If, on the other hand, we transmit over a channel which has a parameter that is worse than the threshold then we can not hope to achieve a low probability of error. This is an important insight but from a practical perspective we would like to know how fast the finite-length performance approaches this asymptotic limit. There can be many different ensembles that all have the same asymptotic performance but that might have a substantially different finite-length behavior. Can we predict which one we should choose a priori without having to resort to simulations? The typical convergence of the performance to the asymptotic limit is shown in Fig. 21. The points correspond to simulation results whereas the solid curves correspond to a general scaling conjecture [2]. Roughly speaking, this scaling conjecture states that around the threshold the error probability behaves as follows: Let the channel be parameterized by $\epsilon$ with increasing $\epsilon$ indicating a worsening of the channel. Let $\epsilon_{\mathrm{d}}$ be the BP threshold, and define $z = \sqrt{n}(\epsilon - \epsilon_{\mathrm{d}})$. Then for $z$ fixed and $n$ tending to infinity we have

$$\mathrm{P_B}(n, \epsilon) = \Phi\left(z/\alpha\right)\big(1 + o(1)\big),$$

where $\Phi(\,\cdot\,)$ is the error function (i.e. $\Phi(x)$ is the probability that a standard normal random variable is smaller than $x$), and $\alpha$ is a constant which depends on the channel as well as on the channel.

For the BEC this scaling law has been shown to be correct [2]. In fact, even a refined version is known [13]: define $z = \sqrt{n}(\epsilon - \epsilon_{\mathrm{d}} + \beta n^{-\frac{2}{3}})$ where $\beta$ is a constant depending on the ensemble. Then for $z$ fixed and $n$ tending to infinity we have

$$\mathrm{P_B}(n, \epsilon) = Q\left(z/\alpha\right)\big(1 + O\big(n^{-1/3}\big)\big).$$

For general channels on the other hand the problem is largely open. If proved to be correct, finite length scaling laws could be used as a tool for an efficient finite-length optimization.

## 7.3 Capacity-Achieving Codes

For the most part we have taken the point of view that we are given an ensemble of codes and a family of channels and we would like to determine the performance of this combination. For instance, the most fundamental question is to determine the threshold noise for such a code/decoder combination. Hopefully this threshold is close to the best possible as determined by Shannon's capacity formula.

But we can take a more active point of view. Given a family of channels how should we choose the ensemble in order for the threshold noise to be as high as possible. In other words, can we approach the capacity of the channel?

For the BSC this question has been answered by Luby, Mitzenmacher, Shokrollahi, Spielman, and Steman in [27]. These authors showed that by a suitable choice of the degree distribution one can

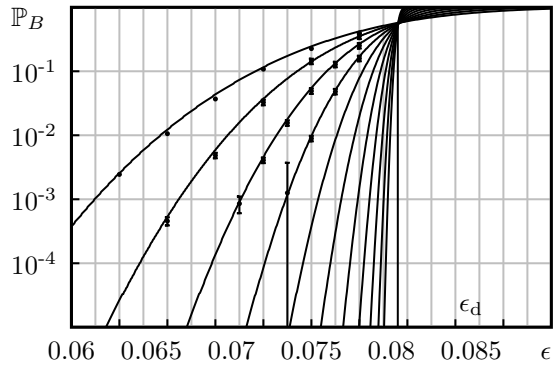


Figure 21: Performance of the BP decoder for the $(3,6)$-regular ensemble when transmission takes place over the BSC. The block lengths are $n = 2^i$, $i = 10, \cdots, 20$. The dots correspond to simulations. For most simulation points the 95% confidence intervals are smaller than the dot size. The lines correspond to the analytic approximation based on scaling laws.

approach the capacity arbitrarily closely. More precisely, in order to approach capacity up to a fraction $\delta$ the average degree has to grow like $\log(1/\delta)$ and this is the best possible. For general channels it is not known whether capacity can be achieved. Although the resolution of this problem will most likely only have a small practical implication it is without doubt the most important open theoretical problem in this area.

# A A Generating Function Calculation

We want to compute the number of ways of picking $n$ distinct objects from $M$ groups each containing $k$ objects, in such a way that the number of selected elements in each group is even. Let us denote this number by $C_n$. The result of this computation is used in Section 3.3, where we claimed the result to be $C_n = \mathsf{coeff}[q_k(z)^M, z^n]$ (in that case $n = lw$).

First notice that given $m_1, \ldots, m_M$ all even, the number of ways of choosing $m_1$ objects from the first group, $m_2$ from the second, etc is

$$\binom{k}{m_1}\binom{k}{m_2}\cdots\binom{k}{m_M}. \tag{A.1}$$

The desired number $C_n$ is the sum of this quantity over the even numbers $m_1, \cdots, m_n$ such that $m_1, \ldots, m_M \in \{0, \ldots k\}$ and $m_1 + \cdots + m_M = n$. The corresponding generating function is

$$C(z) \equiv \sum_{n=0}^{kM} C_n z^n = \sum_{m_1 \ldots m_n}^{\text{even}} \binom{k}{m_1}\binom{k}{m_2}\cdots\binom{k}{m_M} z^{m_1}\cdots z^{m_N}, \tag{A.2}$$

the sum being restricted over even integers $m_1, \ldots, m_M \in \{0, \ldots k\}$. We now notice that the sum factorizes yielding

$$C(z) = \left\{ \sum_{m\text{even}} \binom{k}{m} z^m \right\}^M = q_k(z)^M, \tag{A.3}$$

which proves the claim.

## Acknowledgments

We would like to thank Jean-Philippe Bouchaud and Marc Mézard for organizing the 2006 Summer School on Complex Systems in Les Houches. We would also like to thank Cyril Méasson and Christine Neuberg with their help in creating Fig. 19.

## References

[1] D. Aldous and J. M. Steele, *The Objective Method: Probabilistic Combinatorial Optimization and Local Weak Convergence*, in Probability on discrete structures, H. Kesten, ed., Springer, New York, 2004.

[2] A. Amraoui, A. Montanari, T. Richardson, and R. Urbanke, *Finite-length scaling for iteratively decoded LDPC ensembles*, in Proc. 41th Annual Allerton Conference on Communication, Control and Computing, Monticello, IL, 2003.

[3] O. Barak and D. Burshtein, *Lower bounds on the spectrum and error rate of LDPC code ensembles*, in Proc. of the IEEE Int. Symposium on Inform. Theory, Adelaide, Australia, September 4–9 2005, pp. 42–46.

[4] G. Bauch, H. Khorram, and J. Hagenauer, *Iterative equalization and decoding in mobile communications systems*, in Proceedings of GLOBECOM, 1997.

[5] A. Bennatan and D. Burshtein, *Iterative decoding of LDPC codes over arbitrary discrete-memoryless channels*, in Proceedings of the 41-st Allerton Conference on Communication, Control, and Computing, Monticello, IL, Oct. 2003, pp. 1416–1425.

[6] E. R. Berlekamp, *Algebraic Coding Theory*, Aegean Park Press, 1984.

[7] C. Berrou, A. Glavieux, and P. Thitimajshima, *Near Shannon limit error-correcting coding and decoding*, in Proceedings of ICC'93, Geneve, Switzerland, May 1993, pp. 1064–1070.

[8] H. A. Bethe, *Statistical theory of superlattices*, Proc. Roy. Soc. London A, 150 (1935), pp. 552–558.

[9] R. E. Blahut, *Theory and Practice of Error Control Codes*, Addison-Wesley, 1983.

[10] L. E. Boltzmann, *Vorlesungen über Gastheorie*, J.A. Barth, Leipzig, 1896.

[11] A. Braunstein, M. Mézard, and R. Zecchina, *Survey propagation: algorithm for satisfiability.* arXiv:cond-math/cond-mat/0212002.

[12] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley, New York, 1991.

[13] A. Dembo and A. Montanari, *Finite size scaling for the core of large random hypergraphs.* Xarch:math.PR/0702007, 2007.

[14] C. Douillard, A. Picart, M. Jézéquel, P. Didier, C. Berrou, and A. Glavieux, *Iterative correction of intersymbol interference: Turbo-equalization*, European Trans. on Commun., 6 (1995), pp. 507–511.

[15] G. D. Forney, Jr., *Codes on graphs: Normal realizations*, IEEE Trans. Inform. Theory, 47 (2001), pp. 520–548.

[16] J. Franco and M. Paull, *Probabilistic analysis of the Davis-Putnam procedure for solving satisfiability*, Discrete Appl. Math, 5 (1983), pp. 77–87.

[17] S. Franz, M. Leone, A. Montanari, and F. Ricci-Tersenghi, *Dynamic phase transition for decoding algorithms*, Phys. Rev. E, 22 (2002), p. 046120.

[18] S. Franz and G. Parisi, *Recipes for metastable states in spin glasses*, J. Physique I, 5 (1995), p. 1401.

[19] R. G. Gallager, *Low-density parity-check codes*, IRE Transactions on Information Theory, 8 (1962), pp. 21–28.

[20] ——, *Information theory and reliable communication*, Wiley, 1968.

[21] J. Garcia-Frias and J. D. Villasenor, *Combining hidden Markov source models and parallel concatenated codes*, IEEE Communications Letters, 1 (1997), pp. 111–113.

[22] ——, *Turbo decoders for Markov channels*, IEEE Commun. Lett., 2 (1998), pp. 257–259.

[23] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., New York, 1979.

[24] R. Kikuchi, *A theory of cooperative phenomena*, Phys. Rev., 81 (1951), pp. 988–1003.

[25] F. R. Kschischang and A. W. Eckford, *Low-density parity-check codes for the gilbert-elliot channel*, in Proc. 41th Annual Allerton Conference on Communication, Control and Computing, Monticello, IL, 2003.

[26] S. Lin and D. J. Costello, Jr., *Error Control Coding*, Prentice-Hall, 2nd ed., 2004.

[27] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. A. Spielman, and V. Stemann, *Practical loss-resilient codes*, in Proceedings of the 29th annual ACM Symposium on Theory of Computing, 1997, pp. 150–159.

[28] D. J. C. MacKay, *Good error correcting codes based on very sparse matrices*, IEEE Trans. Inform. Theory, 45 (1999), pp. 399–431.

[29] D. J. C. MacKay, *Information Theory, Inference & Learning Algorithms*, Cambridge University Press, Cambridge, 2002.

[30] D. J. C. MacKay and R. M. Neal, *Good codes based on very sparse matrices*, in Cryptography and Coding. 5th IMA Conference, C. Boyd, ed., no. 1025 in Lecture Notes in Computer Science, Springer, Berlin, 1995, pp. 100–111.

[31] N. Macris, *Correlation inequalities: a useful tool in the theory of LDPC codes*, in Proc. of the IEEE Int. Symposium on Inform. Theory, Adelaide, Australia, Sept. 2005, pp. 2369 – 2373.

[32] E. E. Majani and H. Rumsey, Jr., *Two results on binary-input discrete memoryless channels*, in Proc. of the IEEE Int. Symposium on Inform. Theory, June 1991, p. 104.

[33] E. Maneva, E. Mossel, and M. J. Wainwright, *A new look at survey propagation and its generalizations*, in SODA, Vancouver, Canada, 2005.

[34] R. J. McEliece, *Are turbo-like codes effective on nonstandard channels?*, IEEE inform. Theory Soc. Newslett., 51 (2001), pp. 1–8.

[35] C. Méasson, A. Montanari, T. Richardson, and R. Urbanke, *The Generalized Area Theorem and Some of its Consequences*. submitted to IEEE IT, 2005.

[36] M. Mezard and A. Montanari, *Information, Physics and Computation*, Clarendon Press - Oxford, 2007. to be published.

[37] M. Mézard and G. Parisi, *The bethe lattice spin glass revisited*, Eur. Phys. J. B, 20 (2001), p. 217.

[38] A. Montanari, *The glassy phase of Gallager codes*, Eur. Phys. J. B, 23 (2001), pp. 121–136.

[39] ——, *Tight bounds for LDPC and LDGM codes under MAP decoding*, IEEE Trans. Inform. Theory, 51 (2005), pp. 3221–3246.

[40] A. Montanari and D. Shah, *Counting good truth assignments of random k-sat formulae*, in SODA, New Orleans, USA, Jan. 2007, pp. 1255–1264.

[41] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, Cambridge, 1995.

[42] T. Murayama, Y. Kabashima, D. Saad, and R. Vicente, *Statistical physics of regular low-density parity-check error-correcting codes*, Phys. Rev. E, 62 (2000), p. 1577.

[43] C. Neuberg, *Gilbert-Elliott channel and iterative decoding*. EPFL, Semester Project (Supervisor: Cyril Méason), 2004.

[44] H. Nishimori, *Statistical Physics of Spin Glasses and Information Processing*, Oxford University Press, Oxford, 2001.

[45] J. Pearl, *Fusion, propagation, and structuring in belief networks*, Artificial Intelligence, 29 (1998), pp. 241–288.

[46] P. W. A. R. Abou-Chacra and D. J. Thouless, *A selfconsistent theory of localization*, J. Phys C, 6 (1973), p. 1734.

[47] V. Rathi, *On the asymptotic weight and stopping set distribution of regular LDPC ensembles*, IEEE Trans. Inform. Theory, 52 (2006), pp. 4212–4218.

[48] T. Richardson and H. Jin, *A new fast density evolution*, in Proc. of the IEEE Inform. Theory Workshop, Monte Video, Uruguay, Feb. 2006. pp. 183–187.

[49] T. Richardson and R. Urbanke, *The capacity of low-density parity check codes under message-passing decoding*, IEEE Trans. Inform. Theory, 47 (2001), pp. 599–618.

[50] ——, *Modern Coding Theory*, Cambridge University Press, 2007. to be published.

[51] P. Rujan, *Finite temperature error-correcting codes*, Phys. Rev. Lett., 70 (1993), pp. 2968–2971.

[52] I. Sason and S. Shamai, *Performance Analysis of Linear Codes under Maximum-Likelihood Decoding: A Tutorial*, vol. 3 of Foundations and Trends in Communications and Information Theory, NOW, Delft, the Netherlands, July 2006.

[53] C. E. Shannon, *A mathematical theory of communication*, Bell System Tech. J., 27 (1948), pp. 379–423, 623–656.

[54] N. Shulman and M. Feder, *The uniform distribution as a universal prior*, IEEE Trans. Inform. Theory, 50 (2004), pp. 1356–1362.

[55] M. Sipser and D. A. Spielman, *Expander codes*, IEEE Trans. Inform. Theory, 42 (1996), pp. 1710–1722.

[56] N. Sourlas, *Spin-glass models as error-correcting codes*, Nature, 339 (1989), pp. 693–695.

[57] S. Tatikonda and M. Jordan, *Loopy belief propagation and Gibbs measures*, in Proc. Uncertainty in Artificial Intell., Alberta, Canada, Aug. 2002, pp. 493–500.

[58] J. van Mourik, D. Saad, and Y. Kabashima, *Critical noise levels for LDPC decoding*, Physical Review E, 66 (2002).

[59] C.-C. Wang, S. R. Kulkarni, and H. V. Poor, *Density evolution for asymmetric memoryless channels*, IEEE Trans. Inform. Theory, 51 (2005), pp. 4216–4236.

[60] G. Wiechman and I. Sason, *On the parity-check density and achievable rates of LDPC codes for memoryless binary-input output-symmetric channels*, in Proc. of the Allerton Conf. on Commun., Control and Computing, Monticello, IL, USA, September 28–30 2005, pp. 1747–1758.

[61] J. S. Yedidia, W. T. Freeman, and Y. Weiss, *Constructing free energy approximations and generalized belief propagation algorithms*, IEEE Trans. Info.Theory, 51 (2005), pp. 2282–2313.