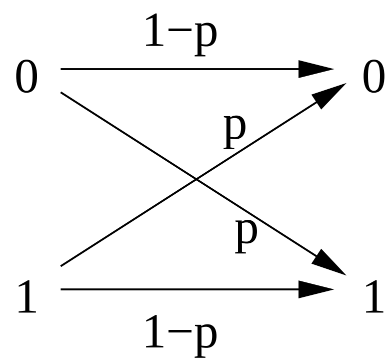
STAT 206A: Gibbs Measures	Fall 2006
Lecture 14	
Lecture date: Oct 12	*Scribe: Alex Fabrikant*

1 Capacity of LDPC codes

As in the preceding lecture, let us define a Binary Symmetric Channel (BSC) with parameter p, which, for each bit of the transmission independently, flips the bit with probability p and transmits it correctly with probability $1 - p$:



Given an LDPC code, we want to ask whether we can decode the output of a BSC transmission encoded by this code.

Claim 1 *Let $p < \frac{1}{2}$. Then the "most" likely codeword x that was sent via a BSC, given that we received word y is given by the $z \in \mathcal{C}$ that minimizes the Hamming distance $d_H(y, z)$*

Proof: By vigorous assertion of obviousness. □

1.1 Bhattacharya Bound

Assume, WLOG, that the zero codeword was sent, and y was received. Let $P_B(\mathcal{C}) = \mathbf{P}[\exists z \in \mathcal{C}, z \neq 0 \text{ s.t. } d(y, z) \leq d(y, 0)]$, i.e. the probability that the decoding will be wrong. For an LDPC ensemble, let $P_B = \mathbf{E}[P_B(\mathcal{C})]$.

Claim 2 *$P_B \leq \sum_{w=1}^{N} \overline{W}(w) e^{-\gamma w}$, where $\gamma = -\log \sqrt{p(1-p)}$ and $\overline{W}$ is the expectation, over the LDPC ensemble, of the number of codewords of weight w. Equivalently, $P_B \leq \sum_{w=1}^{N} \overline{W}(w)(4p(1-p))^{w/2}$.*

Proof: From the definition, we have:

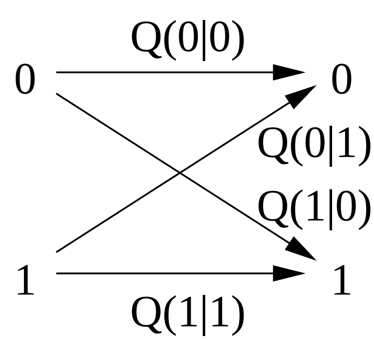
$$
\begin{aligned}
P_B &\leq \mathbf{E}\left[\sum_{0 \neq x \in \mathcal{C}} \mathbf{P}\left[d(x,y) \leq d(0,y)\right] \right] && (1) \\
&= \sum_{w=1}^{N} \overline{W}(w) \mathbf{P}\left[d(x(w),y) \leq d(0,y)\right] && (2)
\end{aligned}
$$

Note that, in the first line, the outer expectation is over the LDPC ensemble, the sum is over codewords in the code, and the probability is over the distribution of y's as received over a BSC. In the second line, $x(w)$ is a weight-w word, which we can set, without loss of generality, to the bit string consisting of w 1's followed by $N-w$ 0's.

$$
\begin{aligned}
\mathbf{P}\left[d(x(w),y) \leq d(0,y)\right] &= \mathbf{P}\left[\exp\left(\lambda(d(0,y) - d(x(w),y))\right) \geq 1\right] \quad (\lambda \geq 0) && (3) \\
&\leq \mathbf{E}\left[\exp\left(\lambda(d(0,y) - d(x(w),y))\right)\right] && (4)
\end{aligned}
$$

This is just equal to the product of these expectations for each bit of y separately (since BSC output bits are flipped independently of each other), yielding $\mathbf{P}[d(x(w),y) \leq d(0,y)] \leq (pe^{\lambda} + (1-p)e^{-\lambda})^w$ for any $\lambda \geq 0$. It can be shown that this bound is strongest for $\lambda = \frac{1}{2}\log\left(\frac{1-p}{p}\right)$, yielding a bound of $(2\sqrt{p(1-p)})^w$, and thus $P_B \leq \sum_{w=1}^{N} \overline{W}(w)(4p(1-p))^{w/2}$.
$\square$

Exercise 3 *Derive a Bhattacharya bound for the general binary memoryless case:*



Hint: use a different distance function, $d_Q(x,y) = -\sum_{i=1}^{N} \log Q(y_i|x_i)$. Instead of $\log\sqrt{4p(1-p)}$, use $\log Q_B$, where $Q_B = \sum_{y \in \{0,1\}} \sqrt{Q(y|1)Q(y|0)}$.

Claim 4 *Consider* $LDPC(\Lambda, P)$ *where* $l_{\min}$, *the minimum degree, is at least 3, and BSC with parameter* p. *Let* $\gamma = -\log\sqrt{4p(1-p)}$. *Suppose that* $\varphi(\rho) < \gamma\rho$ *for all* $\rho \in (0, x)$; $e^{N\varphi(\rho)} = \overline{W}(\rho N)$. *Then,* $P_B \to 0$ *as* N *goes to infinity.*

Proof:

From the above, $P_B = \sum_{w=1}^{N} \overline{W}(w) e^{-\gamma w}$. The problem is with small codewords, so we invoke Claim 1 to split the sum by "weight category":

$$P_B = \sum_{w=1}^{\epsilon N} W(w) e^{-\gamma w} + \sum_{w=\epsilon N}^{N} e^{-\gamma w} e^{\varphi(\frac{w}{N})N}$$

We claimed before (without proof) that for $l_{\min} \geq 3$, the factor graph is a good expander for small sets and therefore the first term is 0 with high probability. The second terms is bounded using the estimate of $\bar{W}$ in terms of φ (which is exact up to sub-exponential terms). □

2 Encoding algorithm

We note that encoding linear codes is computationally "easy". Given an NR-bit input z, where $R < 1$ is the rate of the code, just multiply the code matrix G by it to produce the N-bit codeword Gz. It should be noted that, while multiplying a matrix by a vector can be done in time polynomial with respect to their sizes, from the engineering point of view this is often not enough – as linear time encoding and decoding is desired.

3 Decoding algorithm

Without any noise in the channel, decoding would be a just a matter of solving a system of linear equations, which can be done efficiently by Gaussian elimination. However, this is not robust with respect to noise.

The "easiest" decoding algorithm is the *bit-flipping algorithm*:

Claim 5 *Let* $E(I)$ *be the number of unsatisfied constraints at time* I. $E(I)$ *decreases at each step and if* $E(I) = 0$, *then* $X(I) \in \mathcal{C}$.

Two natural assumptions when we consider decoding are:

```
Receive y
x(0) ← y
while x(I) has bits belonging to more UNSAT constraints than SAT ones do
    X(I + 1) ← X(I) with such a bit flipped
    If can't flip anything, then exit
    I ← I + 1
end while
```

1. Channel assumption: $y_i = x_i \oplus z_i$, where z_i's are independent Bernoulli variables with parameter p
2. Distance Assumption: $d(x, y) < \eta N$

If $\eta > p$, the former assumption implies the latter w.h.p.

Definition 6 *A factor graph is a (ϵ, δ)-expander if subsets U of the set of variable nodes of size bounded by $|U| \le \epsilon N$, it holds that $|\partial U| \ge \delta |U|$, where δU is the set of factor nodes adjacent to at least one node in U.*

Claim 7 *Consider a random (l, k)-regular factor graph $\mathcal{Z}$, where l is the degree of factor nodes, and k — the degree of variable nodes, with $\mathcal{Z}$ thus having N variable nodes and Nl/k factor nodes. Then, for all $\delta \le l - 2$ there exists an $\epsilon > 0$ such that $\mathcal{Z}$ is an (ϵ, δ)-expander with high probability (i.e. probability approaching 1 as N goes to infinity).*

Proof: Left as an exercise. □

Theorem 8 *Consider an (l, k) LDPC on a graph that is an $(\epsilon, \frac{3}{4}l)$-expander. Then the bit-flipping algorithm will correct any pattern of at most $\frac{N\epsilon}{2}$ errors.*

This will be proven in the next lecture.