

We already studied density evolution for the binary erasure channel. The objective of this week is to understand and implement density evolution for a general message passing algorithm.

The material treated this week can be found in Sections 4.4-4.9 of Rüdiger's book, page 218 and afterwards.

General message passing

Recall that a general message passing algorithm is defined by message update rules of the form

$$\nu_{i \rightarrow a}^{(t+1)} = \Phi(y_i; \{\hat{\nu}_{b \rightarrow i}^{(t)} : b \in \partial i \setminus a\}), \quad (1)$$

$$\hat{\nu}_{a \rightarrow i}^{(t)} = \Psi(\{\nu_{j \rightarrow a}^{(t)} : j \in \partial a \setminus i\}). \quad (2)$$

Density evolution for the same algorithm is defined by the distributional recursion

$$\nu^{(t+1)} \stackrel{d}{=} \Phi(y; \hat{\nu}_1^{(t)}, \dots, \hat{\nu}_{l-1}^{(t)}), \quad (3)$$

$$\hat{\nu}^{(t)} \stackrel{d}{=} \Psi(\nu_1^{(t)}, \dots, \nu_{k-1}^{(t)}). \quad (4)$$

Despite the formal similarity between these equations and Eqs. (1), (2), one has to pay attention to the difference. In the density evolution $\stackrel{d}{=}$ denoted equality in distribution between random variables. In these equalities, $\nu^{(t)}$, $\nu_1^{(t)}$, $\nu_2^{(t)}$, \dots are understood to be iid random variables, and the same for $\hat{\nu}^{(t)}$, $\hat{\nu}_1^{(t)}$, $\hat{\nu}_2^{(t)}$, \dots . Further y is distributed according to the transition probability $Q(y|+1)$, l with the variable node degree distribution λ , and k with the check node degree distribution ρ .

A more explicit expression can be written by introducing the distributions of $\nu^{(t)}$ and $\hat{\nu}^{(t)}$ to be denoted respectively as $\mathbf{a}_t(\cdot)$, $\hat{\mathbf{a}}_t(\cdot)$. Assuming for the sake of simplicity that the message alphabet is finite, we can write the density evolution equations as

$$\mathbf{a}_{t+1}(\nu) = \sum_l \lambda_l \sum_y Q(y|+1) \sum_{\hat{\nu}_1 \dots \hat{\nu}_{l-1}} \hat{\mathbf{a}}_t(\hat{\nu}_1) \cdots \hat{\mathbf{a}}_t(\hat{\nu}_{l-1}) \mathbb{I}\{\nu = \Phi(y; \hat{\nu}_1, \dots, \hat{\nu}_{l-1})\}, \quad (5)$$

$$\hat{\mathbf{a}}_t(\hat{\nu}) = \sum_k \rho_k \sum_{\nu_1 \dots \nu_{k-1}} \mathbf{a}_t(\nu_1) \cdots \mathbf{a}_t(\nu_{k-1}) \mathbb{I}\{\hat{\nu} = \Psi(\nu_1, \dots, \nu_{k-1})\}. \quad (6)$$

The asymptotic bit error rate can be computed in terms of the above distributions

$$\lim_{n \rightarrow \infty} \overline{\text{P}}_b(t; n) = \sum_l L_l \sum_y Q(y|+1) \sum_{\hat{\nu}_1 \dots \hat{\nu}_l} \hat{\mathbf{a}}_t(\hat{\nu}_1) \cdots \hat{\mathbf{a}}_t(\hat{\nu}_l) \mathbb{I}\{\hat{x}(y; \hat{\nu}_1, \dots, \hat{\nu}_l) \neq +1\}. \quad (7)$$

Consider a channel family $\text{BMS}(\epsilon)$ ordered by physical degradation with respect to the parameter ϵ . Assuming BP decoding, it is easy to show that the asymptotic bit error rate is non-increasing in t and non-decreasing in ϵ .

Typically, message passing decoding algorithms have a special message value that corresponds to 'maximum reliability.' Let us denote it by $+\infty$. Then the *threshold* for a given message passing decoder and LDPC ensemble is defined as

$$\begin{aligned} \epsilon_* &\equiv \sup \left\{ \epsilon_0 : \lim_{t \rightarrow \infty} \lim_{n \rightarrow \infty} \overline{\text{P}}_b(t; n) = 0 \quad \forall \epsilon \leq \epsilon_0 \right\} \\ &\equiv \sup \left\{ \epsilon_0 : \lim_{t \rightarrow \infty} \mathbf{a}_t = \delta_{+\infty} \quad \forall \epsilon \leq \epsilon_0 \right\}. \end{aligned} \quad (8)$$

For most reasonable message passing decoders, LDPC ensembles, and BMS channels, it is possible to show that ϵ_* is strictly positive.

Implementation

If the message alphabet is finite and its size $|\mathcal{M}|$ is small, one can store $\mathbf{a}_t(\cdot)$, $\hat{\mathbf{a}}_t(\cdot)$ as real vectors and compute them recursively using equations (5), (6).

If the message alphabet is very large or infinite, there are several methods for approximating Eqs. (5), (6) numerically. One that is very easy to program is to replace the distributions $\mathbf{a}(\cdot)$, $\hat{\mathbf{a}}(\cdot)$ by samples

$$\mathfrak{P}_t = \{\nu_1, \nu_2, \dots, \nu_N\}, \quad \hat{\mathfrak{P}}_t = \{\hat{\nu}_1, \hat{\nu}_2, \dots, \hat{\nu}_N\}.$$

These should be thought (ideally) as iid samples with distribution (respectively) $\mathbf{a}_t(\cdot)$, $\hat{\mathbf{a}}_t(\cdot)$. At each time step one generates a new sample $\hat{\mathfrak{P}}_t$ from \mathfrak{P}_t , and a new \mathfrak{P}_{t+1} from $\hat{\mathfrak{P}}_t$, mimicking Eqs. (5), (6).

For instance, to generate the sample $\hat{\mathfrak{P}}_t$, one repeats N times the following operation. Draw an integer k with distribution ρ_k and $(k-1)$ indices $i(1), \dots, i(k-1)$ iid and uniformly random in $[N]$. Then compute a new element of $\hat{\mathfrak{P}}_t$ from the corresponding elements of \mathfrak{P}_t : $\hat{\nu} = \Psi(\nu_{i(1)}, \dots, \nu_{i(k-1)})$.

How do you expect the precision of this algorithm to behave with N ? How would you use it to estimate -say- the bit error rate for a large number of iterations?

There are in fact more efficient ways of implementing density evolution for BP. We will discuss some ideas in class.

WORK !!!!!

Write a program to implement density evolution for a quantized message passing decoder. Use it to determine the threshold of irregular ensembles for the ‘decoder with erasure’ (see last week handout for definition) over the BSC(ϵ).

More precisely, consider the family of ensembles with design rate $r = 1/2$ and degree distribution pair (node perspective)

$$L(x) = (1 - \alpha)x^3 + \alpha x^4, \tag{9}$$

$$R(x) = (1 - \alpha)x^6 + \alpha x^8, \tag{10}$$

parametrized by $\alpha \in [0, 1]$. Determine numerically threshold value $\epsilon_*(\alpha)$ as a function of this parameter.

I expect to receive

1. A print-out of the code.
2. A plot of the curve $\epsilon_*(\alpha)$.