1 ABNNR construction

The ABBNR construction, due to Alon, Brooks, Naor, Naor, and Roth, takes a weak (low-distance) error-correcting code and converts it into a strong (high-distance) code.

In order to construct this code, we will need a d-regular bipartite expander graph $B = (L, R, E)$, where $|L| = |R| = n$.

For any $x = (x_1, \ldots, x_n) \in \mathbb{F}_2^n$, label the vertices in L with $x_1, \ldots, x_n$, in order. Then, each vertex $j \in R$ is adjacent to d vertices in L, which each have a corresponding label x_i. Then, associate with vertex j the sequence of all such x_i, which we call y_j. Note that $y_j \in \mathbb{F}_2^d$, so let $y = (y_1, \ldots, y_n) \in (\mathbb{F}_2^d)^n$.

Note that the map which takes x to y is not itself a good error-correcting code, since if x has a single 1 then y will only have d nonzero entries. Instead, we will pick x from $C_0 \subset \mathbb{F}_2^n$, where C_0 is a code with modest distance. For example, we can fix an explicit linear code C_0 with parameters $\delta(C_0) = 0.01, R(C_0) = 0.5$. Picking d to be very large (much larger than $1/\delta(C_0)$, the code is then the set of all y corresponding to $x \in C_0$. To summarize, the ABBNR code is the composition of two maps:

$$m \in \mathbb{F}_2^{0.5n} \xrightarrow{C_0} x \in \mathbb{F}_2^n \xrightarrow{\text{ABNNR}} y \in (\mathbb{F}_2^d)^n$$

This code has rate $R = 0.5/d$.

What is the distance of this code? This is a linear code, so we can bound the number of zeroes in any codeword. Suppose that for some m, x, y, we have that y has zeroes in all indices corresponding to vertices $D \subset R$. Then, all neighbors of vertices in D have label equal to zero. But since C_0 is a code with relative distance 0.01, this means that the neighborhood $\Gamma(D)$ has size at most $0.99n$.

Exercise 1. *Show that one can pick B to be an appropriate expander graph so that if $|\Gamma(D)| < 0.99n$, then $|D|$ is at most $(1/d + \varepsilon)n$, for any desired ε.*

Thus, $|D|$ is at most roughly n/d, so any nonzero codeword has at most n/d zeroes. Therefore, the code has relative distance approximately $1 - 1/d$.

Therefore, the ABNNR construction gives a $[n, 0.5n/d, (1 - 1/d)n]_{2^d}$-code. Note that if we pick C_0 to have a better rate (since the distance could have been any positive constant), we can push the rate of the ABNNR code to be arbitrarily close to n/d.

Exercise 2. *Show that by concatenating with an appropriate binary code, the ABNNR code can be used to produce a strongly explicit $[n, k, (1/2 - \varepsilon)n]_2$-code, where $n = O(k/\varepsilon^3)$.*

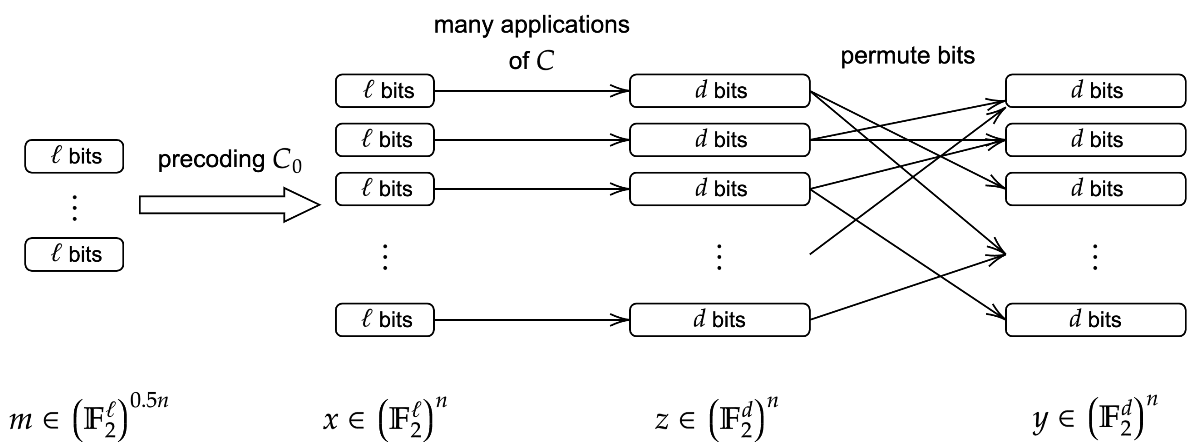
2 AEL codes

The ABNNR codes have good distance, but have the drawback fo having small rate. AEL codes are a generalization of ABNNR codes that have good rates.

We can view the second step of the ABBNR code as first encoding each bit of x with the repetition code (which takes $0, 1$ to $0^d, 1^d$), and then permuting the resulting bits according to a fixed permutation in order to get y. We will generalize to use codes other than repetition codes.

Suppose that we have some linear $[d, \ell, \delta d]_2$-code C; this will be the analog of the repetition code. Then, if we have $x \in (\mathbb{F}_2^\ell)^n$, we obtain $z \in (\mathbb{F}_2^d)^n$ by applying C to each of the elements of x. Then, obtain $y \in (\mathbb{F}_2^d)^n$ by permuting the bits in z according to some fixed permutation. Note that we can form a bipartite graph $B = (L, R, E)$ whose vertices are the elements of z and y, and with an edge between elements that share a bit (i.e., where some two bits map to each other via the permutation).

Finally, to ensure that x has enough nonzero elements, we will have a precoding step as before. Let C_0 be a linear code on the alphabet $\mathbb{F}_2^\ell$ with rate 0.5 and relative distance 0.01. Then, we obtain x by encoding a message $m \in (\mathbb{F}_2^\ell)^{0.5n}$ according to C_0. Then the final AEL code is formed by the map from m to y, as illustrated in the following diagram.



This code has rate $0.5\ell/d$, but as before, by picking a better code C_0 we can push this to $(1 - o(1))\ell/d$, which is roughly the original rate of C.

To analyze the distance of the AEL code, consider some nonzero m, x, z, y; we will bound the number of zeroes in y. Again let $D \subset R$ be the set of all vertices of B whose corresponding elements of y are nonzero. Now, note that at least $0.01n$ elements of x are nonzero, so at least $0.01n$ elements of z have at most $(1-\delta)d$ zero bits. These elements can have at most $(1 - \delta)d$ neighbors in D, since otherwise they must share a nonzero bit with D.

Thus, defining $\Gamma_{\leq(1-\delta)d}(D)$ to be the set of vertices in L that have at most $(1 - \delta)d$ neighbors in D, we must have $|\Gamma_{\leq(1-\delta)d}(D)| \geq 0.01n$. However, we can pick B so that any such D has size at most roughly $(1 - \delta)n$, and permute the bits in any manner consistent with B. Then, y can have at most $(1 - \delta)n$ zeroes, so the AEL code has distance roughly δn.

Exercise 3. *Fill in the details of picking B in the above distance analysis.*

Thus, the AEL code allows us to get a code with the same rate and relative distance as C, but with much larger n, at the expense of having a larger alphabet.

Guruswami and Indyk later modified this construction slightly to create a linear-time decodable code with the same parameters.