EE/Ma 127c Error-Correcting Codes - Homework Assignment 4

Ling Li, ling@cs.caltech.edu

May 17, 2001

4.1 For input symbol u, the repeating symbol is $\mathbf{v} = (v_1, v_2, \ldots, v_q) = (\underbrace{u, u, \ldots, u}_{q})$. In the junction tree of u, $(u, \mathbf{v})$, and v_i $(i = 1, \ldots, q)$,

- the message from u to $(u, \mathbf{v})$ is $\pi(u)$, where $\pi(u)$ is the *a priori* probability of u. However, we will use the *a priori* log-likelihood for u:
$$\mathrm{LLR}_u^{(i)} = \log \frac{\pi(u=0)}{\pi(u=1)}.$$
- The message from v_i to $(u, \mathbf{v})$ is (y_i^e is the channel observed output of v_i)
$$\mathrm{LLR}_i^{(o)} = \log \frac{\Pr\{v_i = 0 | y_i^e\}}{\Pr\{v_i = 1 | y_i^e\}}.$$
- The message from $(u, \mathbf{v})$ to u is
$$\sum_{\mathbf{v}} \chi(u, \mathbf{v}) \prod_i \Pr\{v_i | y_i^e\} = \prod_i \Pr\{v_i = u | y_i^e\},$$
where $\chi(u, \mathbf{v})$ is the local kernel for $(u, \mathbf{v})$ and $\chi(u, \mathbf{v}) = 1$ iff $\mathbf{v}$ is the repetition codeword of u. The log-likelihood version of the message is
$$\sum_i \mathrm{LLR}_i^{(o)}.$$
- The message from $(u, \mathbf{v})$ to v_i is
$$\sum_{j:j\neq i} \mathrm{LLR}_j^{(o)} + \mathrm{LLR}_u^{(i)}.$$

Thus

(a) An efficient APP decoding rule for the information bit is
$$\mathrm{APP}_u = \mathrm{LLR}_u^{(i)} + \sum_i \mathrm{LLR}_i^{(o)},$$
i.e., sum of the *a priori* log-likelihood and the extrinsic information. And the rule for encoded bit v_i is
$$\mathrm{APP}_{v_i} = \mathrm{LLR}_i^{(o)} + \sum_{j:j\neq i} \mathrm{LLR}_j^{(o)} + \mathrm{LLR}_u^{(i)} = \sum_j \mathrm{LLR}_j^{(o)} + \mathrm{LLR}_u^{(i)}.$$
It is not surprising that $\mathrm{APP}_u = \mathrm{APP}_{v_i}$, since the encoding forces $v_i = u$.

(b) For the $(6,2)$ code, we have

$$
\begin{aligned}
\mathrm{APP}_{u_1} &= \mathrm{LLR}_1^{(i)} + \sum_{i=1}^{3} \mathrm{LLR}_i^{(o)} = 0.4, \\
\mathrm{APP}_{u_2} &= \mathrm{LLR}_2^{(i)} + \sum_{i=4}^{6} \mathrm{LLR}_i^{(o)} = -0.4,
\end{aligned}
$$

and $\mathrm{APP}_{v_i} = \mathrm{APP}_{u_1} = 0.4$ for $i = 1,2,3$, and $\mathrm{APP}_{v_i} = \mathrm{APP}_{u_2} = -0.4$ for $i = 4,5,6$.

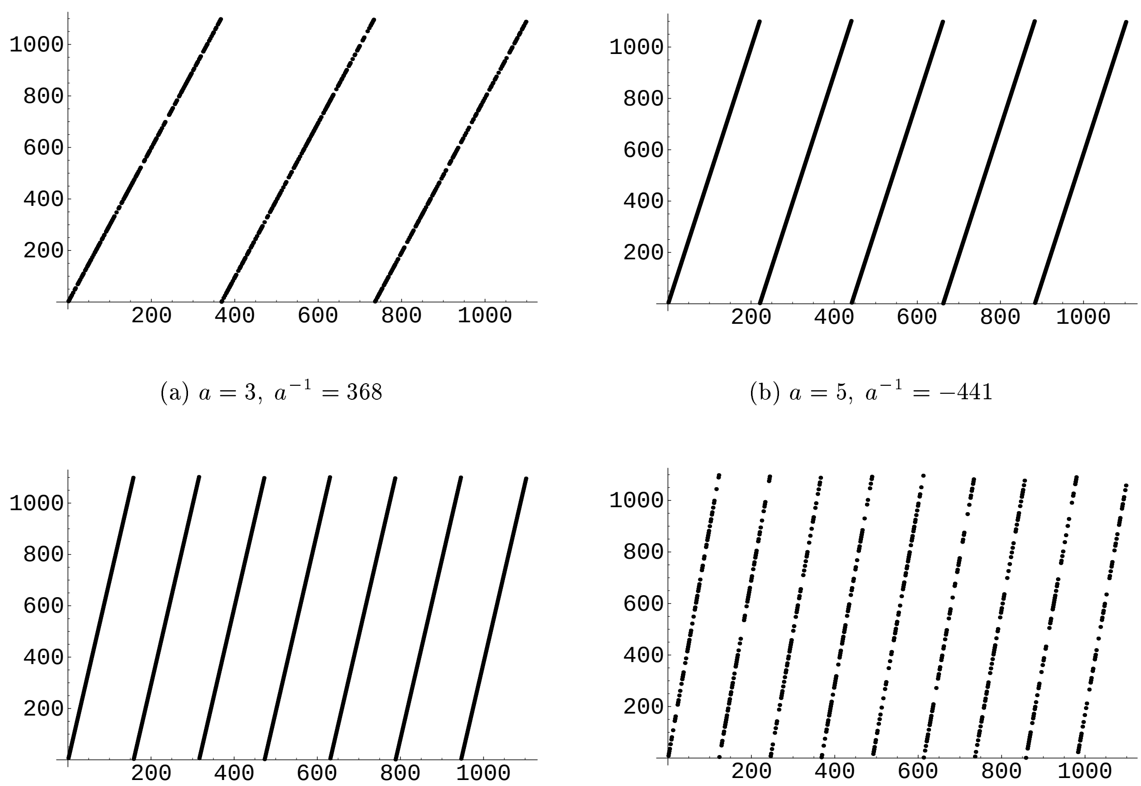
4.2 For $\pi(i) \equiv b \cdot a^i \pmod p$ where p is a prime, we get

$$\pi(i+1) = a\pi(i) \pmod p$$

and

$$\pi(i) = a^{-1}\pi(i+1) \pmod p, \tag{1}$$

where a^{-1} is the inverse of a modulo p. WLOG, assume $|a| < \frac{p}{2}$ and $\left|a^{-1}\right| < \frac{p}{2}$. Thus the point $(\pi(i), \pi(i+1))$ falls on line $y = ax - kp$ for some k. Since π is a permutation, $\pi(i)$ goes



(a) $a = 3,\ a^{-1} = 368$

(b) $a = 5,\ a^{-1} = -441$

(c) $a = 7,\ a^{-1} = -315$

(d) $a = 9,\ a^{-1} = -245$

Figure 1: Plots of $y = ax \pmod p$ where $p = 1103$ and $x = 1, 2, \ldots, p-1$. Produced by Mathematica: `x = Table[{i-1, i}, {i, 1, p-1}]; ListPlot[PowerMod[a, x, p]]`.

over $1, 2, \ldots, p-1$, and k has $|a|$ different values, i.e., $k = 0, 1, \ldots, a-1$ when $a > 0$, and $k = a, a+1, \ldots, -1$ when $a < 0$. Hence the plot $\pi(i+1)$ vs. $\pi(i)$ seems to consist of $|a|$ lines (Figure 1). However, from the viewpoint of (1), all the points fall on lines $x = a^{-1}y - kp$. Thus the plot also seems to consist of $\left|a^{-1}\right|$ lines. Either viewpoint shows that $\pi(i+1)$ and $\pi(i)$ are not independent. (Thus π is not a good random permutation.)

In order to make $\pi(i)$ and $\pi(i+1)$ seem more independent, one idea is to make both $|a|$ and $\left|a^{-1}\right|$ as large as possible. Simple search found that $(a, a^{-1}) = \pm(531, 538)$ are the only two pairs that both $|a|$ and $\left|a^{-1}\right|$ are larger than 530. Figure 2(b) gives the plot for $a = 531$.

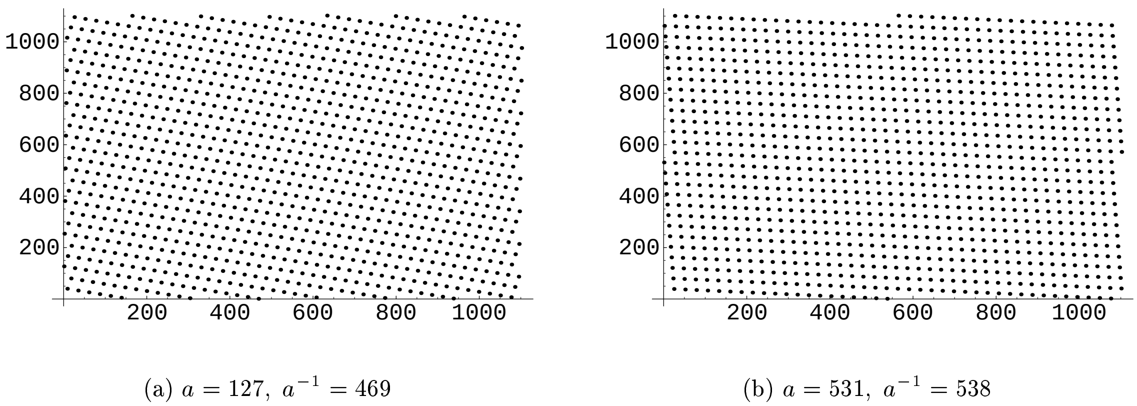


Figure 2: See Figure 1 for detail.

Though it seems similar to $a = 127$ (Figure 2(a)), I guess that $a = 531$ is better than $a = 127$ for an interleaver.

The plot of $\pi(i+2)$ vs. $\pi(i)$ is basically the same as the plot of $\pi(i+2)$ vs. $\pi(i)$, with a^2 instead of a. Thus $\pi(i+2)$ is also dependent on $\pi(i)$.

4.3 Let's denote the information word by $\mathbf{u} = (u_1, \ldots, u_k)$, and the internal word after the interleaver, $\mathbf{v} = (v_1, \ldots, v_{qk})$. Then the codeword is $\mathbf{x} = (x_1, \ldots, x_{qk})$, where $x_1 = v_1$, and $x_{i+1} = x_i + v_{i+1}$ for $i \geq 1$. There is a one-one mapping between $\{\mathbf{v}\}$ and $\{\mathbf{x}\}$. So different mappings from $\mathbf{u}$ to the codeword $\mathbf{x}$ (which are different encoding schemes) make different mapping from $\mathbf{u}$ to $\mathbf{v}$, and their numbers are the same. Remember that there must be exact q u_j's in $\mathbf{v}$ for $j = 1, \ldots, k$. So the number of different mappings from $\mathbf{u}$ to $\mathbf{v}$ is

$$\frac{(qk)!}{(q!)^k}. \tag{2}$$

If we just care about the code, that is, the set $\{\mathbf{x}\}$, then we should divide (2) by $k!$, which is the number of permutations of $u_1, \ldots, u_k$. That is, the number of different (q, k) RA codes is

$$\frac{(qk)!}{k!(q!)^k}.$$