Midterm solutions

Problem M.1 (70 points)

In this problem, we will study a class of codes called product codes.

Suppose that $\mathcal{C}_1$ and $\mathcal{C}_2$ are two binary linear block codes with parameters (n_1, k_1, d_1) and (n_2, k_2, d_2), respectively. We will assume that the first k_1 and k_2 coordinate positions are information sets of $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively.

The product code $\mathcal{C}$ is the code obtained by the following three-step encoding method. In the first step, k_1 independent information bits are placed in each of k_2 rows, thus creating a $k_2 \times k_1$ rectangular array (see Figure 1a). In the second step, the k_1 information bits in each of these k_2 rows are encoded into a codeword of length n_1 in $\mathcal{C}_1$, thus creating a $k_2 \times n_1$ rectangular array (see Figure 1b). In the third step, the k_2 information bits in each of the n_1 columns are encoded into a codeword of length n_2 in $\mathcal{C}_2$, thus creating an $n_2 \times n_1$ rectangular array (see Figure 1c).

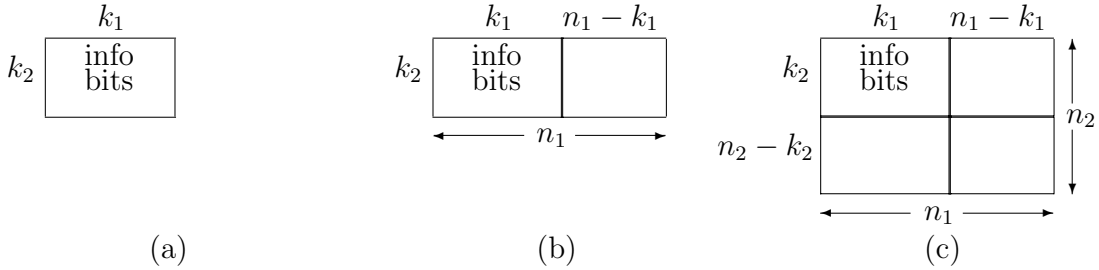

Figure 1. (a) $k_2 \times k_1$ information bit array. (b) $k_2 \times n_1$ array after row encoding. (c) $n_2 \times n_1$ array after column encoding.

(a) Given an (n, k) binary linear block code $\mathcal{C}$, show that the first k coordinate positions are an information set of $\mathcal{C}$ if and only if there exists a generator matrix G for $\mathcal{C}$ whose first k columns form a $k \times k$ identity matrix.

Given an (n, k) binary linear block code $\mathcal{C}$, a set of k coordinates is called an information set of $\mathcal{C}$ if the codewords run through all 2^k possible binary k-tuples in that set of coordinates; *i.e.,* if there is a unique codeword associated with every possible binary k-tuple in that set of coordinates. Thus if the first k coordinate positions are an information set, then for $1 \le j \le k$ there exists a unique codeword $\mathbf{g}_j$ that has a 1 in the jth coordinate position, and a 0 in all of the other first k coordinate positions. The codewords $\{\mathbf{g}_j, 1 \le j \le k\}$ are then obviously a set of k linearly independent n-tuples, so they may be taken as a basis for the (n, k) linear code $\mathcal{C}$; *i.e.,* $\{\mathbf{g}_j, 1 \le j \le k\}$ may be taken as a set of generators for $\mathcal{C}$. If the $\mathbf{g}_j$ are taken as the rows of a $k \times n$ generator matrix G for $\mathcal{C}$, then $\mathcal{C} = \{\mathbf{u}G \mid \mathbf{u} \in (\mathbb{F}_2)^k\}$, and the first k columns of G form a $k \times k$ identity matrix; *i.e., G* is the desired generator matrix, which is called a *systematic* generator matrix.

Conversely, suppose that $\mathcal{C}$ has a systematic generator matrix G, so $\mathcal{C} = \{\mathbf{u}G \mid \mathbf{u} \in (\mathbb{F}_2)^k\}$, where the first k columns of G form a $k \times k$ identity matrix I_k. In other words, $G = [I_k \mid P]$, where P is some $k \times (n-k)$ matrix. Then

$$\mathcal{C} = \{\mathbf{u}G \mid \mathbf{u} \in (\mathbb{F}_2)^k\} = \{\mathbf{u}[I_k \mid P] \mid \mathbf{u} \in (\mathbb{F}_2)^k\} = \{(\mathbf{u}, \mathbf{u}P) \mid \mathbf{u} \in (\mathbb{F}_2)^k\}.$$

Thus as $\mathbf{u}$ runs through the set $(\mathbb{F}_2)^k$ of all binary k-tuples, the codewords of $\mathcal{C}$ run through all 2^k possible binary k-tuples in the first k coordinate positions, so the first k coordinate positions form an information set for $\mathcal{C}$. □

Note that the encoding method given above is well-defined if and only if the first k_1 and k_2 coordinate positions are information sets of $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively.

(b) Show that the encoding method given above produces the same codeword whether the encoder encodes first rows and then columns, or first columns and then rows.

It is obvious that for the same initial $k_1 k_2$ information bits in the array of Figure 1a, the same encoded bits are produced in the top and left quadrants of Figure 1c. Therefore the assertion needs to be proved only for encoded bits in the lower right quadrant.

Let us therefore look at a particular encoded bit in the lower right quadrant, say the encoded bit $b_{\ell m}$ in row ℓ and column m. Let $G_1 = [I_{k_1} \mid P]$ be a systematic generator matrix for $\mathcal{C}_1$, and let $G_2 = [I_{k_2} \mid Q]$ be a systematic generator matrix for $\mathcal{C}_2$. Let the information bits in the top left quadrant be denoted by $u_{ij}, 1 \le i \le k_2, 1 \le j \le k_1$. Then an encoded bit b_{im} in the top right quadrant is given by

$$b_{im} = \sum_{j=1}^{k_1} u_{ij} P_{jm}, \quad 1 \le i \le k_2, 1 \le m \le n_1 - k_1.$$

Similarly, an encoded bit in the bottom left quadrant is given by

$$b_{\ell j} = \sum_{i=1}^{k_2} Q_{\ell i} u_{ij}, \quad 1 \le \ell \le n_2 - k_2, 1 \le j \le k_1.$$

Thus if we encode first row-wise and then column-wise, then the encoded bit $b_{\ell m}$ is formed as the linear combination

$$b_{\ell m} = \sum_{i=1}^{k_2} Q_{\ell i} b_{im} = \sum_{i=1}^{k_2} \sum_{j=1}^{k_1} Q_{\ell i} u_{ij} P_{jm}, \quad 1 \le \ell \le n_2 - k_2, 1 \le m \le n_1 - k_1.$$

On the other hand, if we encode first column-wise and then row-wise, then the encoded bit $b_{\ell m}$ is formed as the linear combination

$$b_{\ell m} = \sum_{j=1}^{k_1} b_{\ell j} P_{jm} = \sum_{j=1}^{k_1} \sum_{i=1}^{k_2} Q_{\ell i} u_{ij} P_{jm}, \quad 1 \le \ell \le n_2 - k_2, 1 \le m \le n_1 - k_1.$$

Thus $b_{\ell m}$ is the same under either encoding method. □

A quicker way of showing this is to use matrix notation, including transposes. Then, writing the information array as U, we have

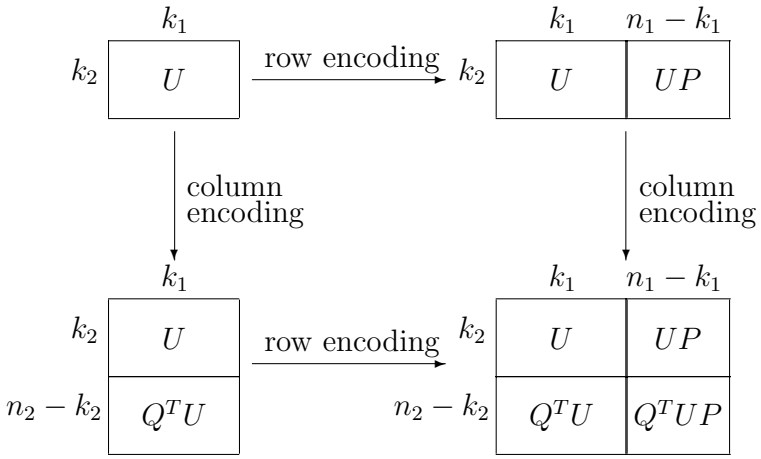

Figure 2. Proof that row/column and column/row encoding produce the same result.

Note that this result implies that if an $n_2 \times n_1$ array is a codeword of $\mathcal{C}$, then every row is a codeword of $\mathcal{C}_1$ and every column is a codeword of $\mathcal{C}_2$.

(c) Show that the product code $\mathcal{C}$ is an $(n_1 n_2, k_1 k_2, d_1 d_2)$ binary linear block code.

The proof of (b) shows that every bit in a codeword is a fixed linear function of the $k_1 k_2$ information bits $u_{ij}, 1 \le i \le k_2, 1 \le j \le k_1$. Therefore the sum of the two codewords corresponding to two information bit arrays is the codeword corresponding to the sum of the two given information bit arrays. Thus $\mathcal{C}$ has the group property; *i.e.,* $\mathcal{C}$ is linear.

Obviously $\mathcal{C}$ is a binary block code of length $n_1 n_2$ bits. Since each information $k_1 k_2$-tuple maps to a distinct codeword in the top left quadrant, the dimension of $\mathcal{C}$ must be $k_1 k_2$.

Since $\mathcal{C}$ is linear, the minimum Hamming distance of $\mathcal{C}$ is the minimum nonzero Hamming weight of any codeword in $\mathcal{C}$. Now if there is any nonzero bit in a codeword of $\mathcal{C}$, then there must be at least d_1 nonzero bits in the same row, since every row is a codeword of $\mathcal{C}_1$, and in each of the $\ge d_1$ corresponding columns, there similarly must be at least d_2 nonzero bits. Therefore any nonzero codeword has weight at least $d_1 d_2$.

To show that the minimum nonzero weight is precisely $d_1 d_2$, consider a weight-d_1 codeword $\mathbf{c} = \{c_i\} \in \mathcal{C}_1$ and a weight-d_2 codeword $\mathbf{c}' = \{c'_j\} \in \mathcal{C}_2$. Then the (tensor product) array $\mathbf{c} \otimes \mathbf{c}'$ with bits $b_{ij} = c_i c'_j$ is a codeword of $\mathcal{C}$ and has weight $d_1 d_2$. In other words, to construct a minimum-weight codeword, write down a minimum-weight codeword in $\mathcal{C}_1$ in the top row of the $n_2 \times n_1$ array. For each column, if the top entry is a 1 then write down a minimum-weight codeword in $\mathcal{C}_2$ which has its first bit equal to 1 in that column. If the top entry is a 0 then simply fill the column with all 0s. The resulting codeword has a weight of precisely $d_1 d_2$. □

(d) Express the nominal coding gain $\gamma_c(\mathcal{C})$ of the Euclidean-space image $s(\mathcal{C})$ of $\mathcal{C}$ in terms of the nominal coding gains $\gamma_c(\mathcal{C}_1)$ and $\gamma_c(\mathcal{C}_2)$ of the Euclidean-space images $s(\mathcal{C}_1)$ and $s(\mathcal{C}_2)$ of $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. Express the nominal spectral efficiency $\rho(\mathcal{C})$ of $\mathcal{C}$ in terms of the nominal spectral efficiencies $\rho(\mathcal{C}_1)$ and $\rho(\mathcal{C}_2)$ of $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively.

The nominal coding gain of the Euclidean-space image $s(\mathcal{C})$ of an (n, k, d) binary linear block code $\mathcal{C}$ is $\gamma_c(\mathcal{C}) = kd/n$, and its nominal spectral efficiency is $\rho(\mathcal{C}) = 2k/n$. Thus

$$\gamma_c(\mathcal{C}) = \frac{k_1 k_2 d_1 d_2}{n_1 n_2} = \frac{k_1 d_1}{n_1}\frac{k_2 d_2}{n_2} = \gamma_c(\mathcal{C}_1)\gamma_c(\mathcal{C}_2),$$

and

$$\rho(\mathcal{C}) = 2\frac{k}{n} = 2\frac{k_1}{n_1}\frac{k_2}{n_2} = \frac{1}{2}\rho(\mathcal{C}_1)\rho(\mathcal{C}_2).$$

(e) Starting with Reed-Muller codes of lengths less than 64, is it possible to use the product code construction to construct a product code of length 64 that has better parameters $(64, k, d)$ than the corresponding RM code of length 64?

The answer is NO, as a few examples quickly show. For example, the product of an $(8, 4, 4)$ code with itself yields a product code with parameters $(64, 16, 16)$, which is inferior to the $(64, 22, 16)$ RM code.

An exhaustive proof is given by the tables below, plus the observation that the product of a trivial code with any code is trivial, and the product of the $(1, 1, 1)$ code with any code gives the latter code again. Note that we do need to check $(2^m, 2^m, 1)$ codes.

×	(32, 1, 32)	(32, 6, 16)	(32, 16, 8)	(32, 26, 4)	(32, 31, 2)	(32, 32, 1)
(2,2,1)	(64, 2, 32)	(64, 12, 16)	(64, 32, 8)	(64, 52, 4)	(64, 62, 2)	(64, 64, 1)
(2,1,2)	(64, 1, 64)	(64, 6, 32)	(64, 16, 16)	(64, 26, 8)	(64, 31, 4)	(64, 32, 2)

×	(16, 1, 16)	(16, 5, 8)	(16, 11, 4)	(16, 15, 2)	(16, 16, 1)
(4,4,1)	(64, 4, 16)	(64, 20, 8)	(64, 44, 4)	(64, 60, 2)	(64, 64, 1)
(4,3,2)	(64, 3, 32)	(64, 15, 16)	(64, 33, 8)	(64, 45, 4)	(64, 48, 2)
(4,1,4)	(64, 1, 64)	(64, 5, 32)	(64, 11, 16)	(64, 15, 8)	(64, 16, 4)

×	(8, 1, 8)	(8, 4, 4)	(8, 7, 2)	(8, 8, 1)
(8,8,1)	(64, 8, 8)	(64, 32, 4)	(64, 56, 2)	(64, 64, 1)
(8,7,2)	(64, 7, 16)	(64, 28, 8)	(64, 49, 4)	(64, 56, 2)
(8,4,4)	(64, 4, 32)	(64, 16, 16)	(64, 28, 8)	(64, 32, 4)
(8,1,8)	(64, 1, 64)	(64, 4, 32)	(64, 7, 16)	(64, 8, 8)

We see that the best results are

$$\begin{aligned}
(32, 32, 1) \times (2, 2, 1) &= (64, 64, 1) = (64, 64, 1);\\
(32, 31, 2) \times (2, 2, 1) &= (64, 62, 2) < (64, 63, 2);\\
(32, 26, 4) \times (2, 2, 1) &= (64, 52, 4) < (64, 57, 4);\\
(16, 11, 4) \times (4, 3, 2) &= (64, 33, 8) < (64, 42, 8);\\
(32, 16, 8) \times (2, 1, 2) &= (64, 16, 16) < (64, 22, 16);\\
(32, 6, 16) \times (2, 1, 2) &= (64, 6, 32) < (64, 7, 32);\\
(32, 1, 32) \times (2, 1, 2) &= (64, 1, 64) = (64, 1, 64),
\end{aligned}$$

which never improve on the corresponding (equal-d) RM codes of length 64, and in fact are worse in all nontrivial cases.

The following neat inductive proof was suggested by Sheng Jing. The product code $\mathrm{RM}(r_1, m_1) \times \mathrm{RM}(r_2, m_2)$ has length $n = 2^{m_1+m_2}$ and minimum distance $d = 2^{m_1+m_2-r_1-r_2}$, so it should be compared with the Reed-Muller code $\mathrm{RM}(r_1 + r_2, m_1 + m_2)$; *i.e.,* $k(r_1, m_1) \times k(r_2, m_2)$ should be compared with $k(r_1 + r_2, m_1 + m_2)$. Suppose that $k(r_1, m_1) \times k(r_2, m_2) \leq k(r_1 + r_2, m_1 + m_2)$ for all lesser (m_1, m_2) and all compatible (r_1, r_2). Then, using $k(r, m) = k(r-1, m-1) + k(r, m-1)$, we can show inductively that

$$\begin{aligned} k(r_1, m_1) \times k(r_2, m_2) &= (k(r_1 - 1, m_1 - 1) + k(r_1, m_1 - 1)) \times k(r_2, m_2) \\ &\leq k(r_1 + r_2 - 1, m_1 + m_2 - 1) + k(r_1 + r_2, m_1 + m_2 - 1) \\ &= k(r_1 + r_2, m_1 + m_2). \end{aligned}$$

(f) Starting with Reed-Muller codes of lengths less than 64, is it possible to obtain a sequence of product codes whose nominal coding gains increase without limit by iterating the product code construction— i.e., by extending the above construction to an m-dimensional product code that maps an array of $k_1 \times k_2 \times \cdots \times k_m$ information bits into $n_1 \times n_2 \times \cdots \times n_m$ binary symbols using binary linear block codes $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_m$? Is it possible to do this while keeping the nominal spectral efficiency above some nonzero value?

Apart from the trivial, universe and repetition codes, the RM codes of lengths less than 64 have nominal coding gains γ_c greater than 1 and rates k/n bounded by $31/32 < 1$. It is therefore possible to iterate the product construction, using any of these codes as components, to obtain a sequence of increasingly long product codes whose nominal coding gains increase without limit.

However, any such sequence must be based on an infinite number of components with rates $k/n \leq 31/32$, and thus the rate k/n and the nominal spectral efficiency $\rho = 2k/n$ of the product code must tend to zero as its coding gain tends to infinity.

(g) The construction of $\mathcal{C}$ suggests the following two-step decoding method. First decode each row, using an optimum (minimum Euclidean distance) decoding method for $\mathcal{C}_1$. This first decoding step yields an array of noisy received bits. Then decode each column, using an optimum (minimum Hamming distance) decoding method for $\mathcal{C}_2$.

Compare the performance and complexity of this two-step decoding method with that of the optimum decoding method on a binary-input AWGN channel. If you like, you may let both $\mathcal{C}_1$ and $\mathcal{C}_2$ be the $(8, 4, 4)$ RM code. As a figure of merit for performance, you may use the minimum squared norm of any error sequence that can cause a decoding error.

On a binary-input AWGN channel, an (n, k, d) binary linear block code $\mathcal{C}$ maps under the standard 2-PAM map $s : \{0, 1\} \rightarrow \{\pm\alpha\}$ to a subset of 2^k vertices of an n-cube of side 2α. The minimum squared distance between two n-vectors in the Euclidean image $s(\mathcal{C})$ is $d^2_{\min} = 4\alpha^2 d$. An optimum decoder is a minimum-distance decoder. Since $d_{\min} = 2\alpha\sqrt{d}$, the minimum norm of any error vector that can cause a decoding error is $d_{\min}/2 = \alpha\sqrt{d}$, so the minimum squared norm of any error-causing vector is $\alpha^2 d$. The decoding complexity of a minimum-distance decoder is of the order of 2^k, since an MD decoder must compute the distance between the received vector and each of the 2^k codewords.

The optimum decoder for an $(n_1 n_2, k_1 k_2, d_1 d_2)$ product code $\mathcal{C}$ is therefore a minimum-distance decoder. The minimum squared norm of any error-causing vector is $\alpha^2 d_1 d_2$, and the decoding complexity is of the order of $2^{k_1 k_2}$. For the $(64, 16, 16)$ product code, the performance figure of merit is $16\alpha^2$, and the decoding complexity is of the order of 2^{16}.

The first step of the suggested suboptimum two-step decoding method consists of decoding each of the n_2 rows using an optimum decoder for $\mathcal{C}_1$. In each row, the performance figure of merit is $d_1\alpha^2$, or $4\alpha^2$ in our example, and the decoding complexity is of the order of 2^{k_1}, or 16 in our example.

The second step of the suggested suboptimum two-step decoding method is to decode each of the n_1 columns, using the results of the first decoding step as noisy binary inputs. We may take the decoding complexity as of the order of 2^{k_2} for each column (although there are usually more efficient near-optimum decoding methods). In each column, an decoding error can be made if there are $d_2/2$ binary errors (assuming d_2 is even).

Therefore an overall product code decoding error can be made if $d_2/2$ rows are in error. By considering the weight-$d_1 d_2$ tensor product of two minimum-weight component codewords as in the proof of minimum distance in part (c), we can see that an error *will* be made in certain such cases. The minimum error squared norm for this to happen is $(d_2/2)d_1\alpha^2$, which is a factor of 2 (3 dB) worse than the optimum decoding method.

The overall decoding complexity is of the order of $n_2 2^{k_1} + n_1 2^{k_2}$. For our example, this yields $2(8 \times 16) = 2^8$, an improvement of the order of a factor of 2^8 over the complexity of the optimum decoder. For long codes, there will be an exponential improvement, of the order of a factor of $2^{k_1 k_2 - \max\{k_1, k_2\}}$. □

(h) [Optional; extra credit] Propose a two-step decoding method that has same figure of merit for performance as optimal decoding, but has decoding complexity similar to that of the suboptimal two-step method proposed in part (g).

The 3 dB performance loss in the suboptimal two-step method of part (g) is at least partly due to making hard decisions in the first decoding step. It would be better to attach some sort of reliability metric to each decoded codeword.

An excellent reliability metric for the decoded bit b_{ij} in the ith row and jth column resulting from the first step of row-wise decodings is the difference $d_{ij}^2 = ||\mathbf{e}_{ij1}||^2 - ||\mathbf{e}_{ij0}||^2$ between the squared norms of the apparent errors $\mathbf{e}_{ij0} = \mathbf{r}_i - s(\mathbf{c}_{ij0})$ and $\mathbf{e}_{ij1} = \mathbf{r}_i - s(\mathbf{c}_{ij1})$, where $\mathbf{r}_i$ is the received vector in the ith row, and $\mathbf{c}_{ij0}$ (resp. $\mathbf{c}_{ij1}$) is the codeword in $\mathcal{C}_1$ such that $s(\mathbf{c}_{ij0})$ (resp. $s(\mathbf{c}_{ij1})$) is the closest codeword image to $\mathbf{r}_i$ such that $\mathbf{c}_{ij0}$ has a 0 (resp. 1) in the jth column. Note that $\mathbf{c}_{ij0}$ and $\mathbf{c}_{ij1}$ must differ in at least d_1 positions. Note also that

$$d_{ij}^2 = ||\mathbf{e}_{ij1}||^2 - ||\mathbf{e}_{ij0}||^2 = 2\langle \mathbf{r}, s(\mathbf{c}_{ij0}) - s(\mathbf{c}_{ij1}) \rangle,$$

since $||s(\mathbf{c})||^2 = n_1\alpha^2$ for all $\mathbf{c} \in \mathcal{C}_1$. Thus if the hard decision bit is $b_{ij} = 0$, *i.e.,* the closest codeword to $\mathbf{r}_i$ has a 0 in the jth column, then $d_{ij}^2 \geq 0$; if $b_{ij} = 1$, then $d_{ij}^2 \leq 0$.

Such a reliability metric may be used in column decoding as follows. To decode the jth column, we look for the column codeword $\mathbf{c}_j \in \mathcal{C}_2$ that maximizes the following sum of sign-weighted reliability metrics: $\sum_i^{n_2} (-1)^{c_{ij}} d_{ij}^2$. In other words, we add the reliability metrics of the hard decision bits b_{ij} in that column, multiplied by a sign term $(-1)^{c_{ij}}$

such that $(-1)^{c_{ij}} d_{ij}^2 \geq 0$ if the codeword bit c_{ij} matches the hard decision bit b_{ij}, and $(-1)^{c_{ij}} d_{ij}^2 \leq 0$ if they do not match, and choose the $\mathbf{c}_j \in \mathcal{C}_2$ that maximizes this sum.

It is easy to see that this decoding algorithm is equivalent to the following one. For each $\mathbf{c}_j \in \mathcal{C}_2$, construct an $n_2 \times n_1$ array by extending each bit c_{ij} in $\mathbf{c}_j$ to the codeword $\mathbf{c}_{ij0}$ or $\mathbf{c}_{ij1}$ in $\mathcal{C}_1$ according to whether $c_{ij} = 0$ or $c_{ij} = 1$. Then compute the Euclidean distance between the Euclidean image of this array and the array of n_2 received rows $\mathbf{r}_i$, and choose the closest as the winner.

In order for a column-j decoding error to occur, the noise must be such that $\mathbf{r}$ is closer to one of these constructed arrays than to the transmitted array. But each of these constructed arrays differs from the transmitted jth column in at least d_2 rows, and in each of those rows differs from the transmitted row in at least d_1 bits. Therefore the minimum squared norm of any error-causing vector is $\alpha^2 d_1 d_2$; *i.e.,* the same as for optimum minimum-distance decoding of the whole product code.

Meanwhile, the decoding complexity of this two-step algorithm is of the order of 2^{k_1} per row and 2^{k_2} per column, or $n_2 2^{k_1} + n_1 2^{k_2} \approx 2^{\max\{k_1,k_2\}}$ in all. □

We conclude that while product codes do not have the best possible parameters (n, k, d), they permit long high-performance codes to be built from short low-complexity component codes, and they admit low-complexity near-optimum decoding algorithms. So product codes might give some hint as to how to construct high-performance codes that can be decoded nearly optimally with reasonable complexity.

Grade distribution on Problem 1 ($N = 16$):
$\{23, 31, 38, 38, 44, 46, 49, 49, 52, 53, 53, 53, 54, 55, 57, 57\}$.

Problem M.2 (30 points)

For each of the propositions below, state whether the proposition is true or false, and give a proof of not more than a few sentences, or a counterexample. No credit will be given for a correct answer without an adequate explanation.

(a) A signal constellation $\mathcal{A}$ consisting of a subset of 2^k points of the 2^n vertices of the n-cube, $k < n$, has a nominal spectral efficiency $\rho(\mathcal{A}) < 2$ b/2D and a nominal coding gain $\gamma_c(\mathcal{A}) \geq 1$.

False. In general, the nominal coding gain may be less than 1. As a counterexample, consider the Euclidean image $\mathcal{A} = s(\mathcal{C})$ of the $(2, 1, 1)$ binary linear block code $\mathcal{C} = \{00, 10\}$. The nominal coding gain of $s(\mathcal{C})$ is $\gamma_c(s(\mathcal{C})) = kd/n = \frac{1}{2}$ (-3 dB).

(b) Let $S = \{a, b, c, d, e, f\}$ be a set of six elements, and let a binary operation $\oplus$ be defined on S by the following "addition table:"

$\oplus$	a	b	c	d	e	f
a	a	b	c	d	e	f
b	b	c	a	f	d	e
c	c	a	b	e	f	d
d	d	e	f	a	b	c
e	e	f	d	c	a	b
f	f	d	e	b	c	a

Then S forms a group under the binary operation $\oplus$. (You need not check the associative law.)

TRUE. Using the alternative group axioms (Theorem 7.1), we need merely check that:

(a) there is an identity element, namely a, since $a \oplus s = s \oplus a = s$ for all $s \in S$;

(b) every row and column is a permutation of the group elements.

Note that this group is nonabelian. In fact, this is the smallest nonabelian group.

(c) Considering the weight distribution of a $(6, 3, 4)$ code over $\mathbb{F}_4$, it is possible that such a code exists.

TRUE. A $(6, 3, 4)$ code over any field is MDS, since $k + d = n + 1$. Therefore, over $\mathbb{F}_4$, the number of words of weight 4 must be

$$N_4 = \binom{6}{4}(q-1) = 15 \times 3 = 45,$$

and the number of words of weight 5 must be

$$N_5 = \binom{6}{5}((q^2-1) - \binom{5}{4}(q-1)) = 6 \times (15-15) = 0.$$

Therefore the number of words of weight 6 must be $N_6 = 63 - 45 = 18$. Since all of these numbers are nonnegative, such a code could exist. □

In fact, such a code does exist, the so-called "hexacode." A systematic generator matrix for one version of the hexacode is

$$\begin{matrix} 1 & 0 & 0 & 1 & \alpha & \alpha \\ 0 & 1 & 0 & \alpha & 1 & \alpha \\ 0 & 0 & 1 & \alpha & \alpha & 1 \end{matrix}$$

It is easy to verify that this $(6, 3)$ code over $\mathbb{F}_4$ has minimum nonzero weight 4 and thus weight distribution $\{1, 0, 0, 0, 45, 0, 18\}$, simply by listing all of its codewords.

GRADE DISTRIBUTION ON PROBLEM 2 ($N = 16$):
$\{0, 6, 13, 15, 15, 15, 16, 18, 20, 21, 23, 23, 23, 24, 26, 27\}$.

GRADE DISTRIBUTION ON MIDTERM EXAM ($N = 16$):
$\{38, 49, 54, 57, 59, 61, 61, 64, 67, 70, 71, 73, 73, 76, 81, 83\}$.