# Chapter 3

# Codes as Images

So far we have only two ways to specify a (linear) code: we can list the codewords, or we can give a spanning set, which might also be a basis. But since codes are vector subspaces, we can also describe them as the image or the kernel of a suitable mapping between spaces. It turns out that these mappings are also helpful for the encoding and decoding processes.

## 3.1 Mappings and Matrices

Let $\mathbb{F}_q^k$ and $\mathbb{F}_q^n$ be two vector spaces, over the same field $\mathbb{F}_q$ but of possibly different dimensions, and let $f : \mathbb{F}_q^k \longrightarrow \mathbb{F}_q^n$ be a linear mapping between them. This means that $f$ preserves the linear structure: if $\mathbf{x}$ and $\mathbf{y}$ are vectors in $\mathbb{F}_q^k$, and $\lambda$ is a scalar in $\mathbb{F}_q$, we have $f(\mathbf{x}+\mathbf{y}) = f(\mathbf{x})+f(\mathbf{y})$ and $f(\lambda\mathbf{x}) = \lambda f(\mathbf{x})$. Then we know that we can perform the mapping $f$ by multiplying by a suitable matrix $A$, with entries from $\mathbb{F}_q$. We are writing our vectors as rows: $\mathbf{x} \in \mathbb{F}_q^k$ is a $1 \times k$ 'matrix', and $f(\mathbf{x}) \in \mathbb{F}_q^n$ is $1 \times n$. So we need $A$ to be $k \times n$; we can write that $A \in M_{k,n}(\mathbb{F}_q)$. And we must multiply $\mathbf{x}$ by $A$ on the *right*.

**Example 24.** Let $f : \mathbb{F}_5^3 \longrightarrow \mathbb{F}_5^4$ such that $f(\mathbf{x}) = \mathbf{x}A$, where $A = \begin{pmatrix} 1 & 0 & 1 & 3 \\ 3 & 2 & 0 & 1 \\ 4 & 2 & 1 & 4 \end{pmatrix} \in M_{3,4}(\mathbb{F}_5)$. Then, for example, if $\mathbf{x} = (0, 1, 2)$,

$$f(\mathbf{x}) = \mathbf{x}A = (0,1,2) \begin{pmatrix} 1 & 0 & 1 & 3 \\ 3 & 2 & 0 & 1 \\ 4 & 2 & 1 & 4 \end{pmatrix} = (1,1,2,4).$$

△

This may look unfamiliar. In Linear Algebra 1 you mostly wrote vectors as columns, and multiplied by $A$ on the left. Of course, we could still do it that way, by taking the transpose of everything: if $\mathbf{x}A = \mathbf{y}$, then also $A^t\mathbf{x}^t = (\mathbf{x}A)^t = y^t$. But coding theory always uses row vectors, and you will get used to it!

The **image** of the mapping $f$ is all the vectors in $\mathbb{F}_q^n$ which can be written as $f(\mathbf{x})$ for some $\mathbf{x}$ in $\mathbb{F}_q^k$:

$$\mathrm{im}(f) = f(\mathbb{F}_q^k) = \{f(\mathbf{x}) \mid \mathbf{x} \in \mathbb{F}_q^k\} = \{\mathbf{x}A \mid \mathbf{x} \in \mathbb{F}_q^k\} \subseteq \mathbb{F}_q^n$$

(Draw your own"fried egg" diagram.) Let us regard the rows of $A$ as vectors $\mathbf{a}_1, \ldots, \mathbf{a}_k \in \mathbb{F}_q^n$. Then since $\mathbf{x} = (x_1, \ldots, x_n)$ can be any vector in $\mathbb{F}_q^k$, $\mathrm{im}(f)$ is all the linear combinations $x_1\mathbf{a}_1 + \cdots + x_k\mathbf{a}_k$ of the $\mathbf{a}_i$. This is the span of the $\mathbf{a}_i$, so we can say that $\mathrm{Im}(f) = \langle\{\mathbf{a}_1, \ldots, \mathbf{a}_k\}\rangle$, or that the rows of $A$ are a spanning set for the image.

## 3.2 Generator-matrices

If the rows of $A$ are also linearly independent, then they are a basis for $\mathrm{Im}(f)$, and $k$, the number of rows, is the dimension of this image.

**Definition 3.1.** For some matrix $G \in M_{k,n}(\mathbb{F}_q)$, let $f : \mathbb{F}_q^k \longrightarrow \mathbb{F}_q^n$ be defined by $f(\mathbf{x}) = \mathbf{x}G$, and let the linear code $C = \mathrm{im}(f) = \{\mathbf{x}G \mid \mathbf{x} \in \mathbb{F}_q^k\} \subseteq \mathbb{F}_q^n$. Then if $G$ has linearly independent rows, G is a **generator-matrix** for C.

So if we are given a code as a span, $C = \langle\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}\rangle$, how can we find a generator-matrix for $C$? We can easily make the matrix $A$ with rows $\mathbf{a}_i$, but we still need to determine whether these rows are linearly independent. If they are not, we must find another, smaller set of vectors which are independent, but span the same subspace of $\mathbb{F}_q^n$.

As you know, this can be done by row-reducing the matrix $A$. This process, which takes linear combinations of the rows, does not change their span. Of course, the row-reduction must be done in $\mathbb{F}_q$, but this is easier than in $\mathbb{R}$. The numbers stay small, and you never need to subtract or divide: just use inverses, and add or multiply.

We obtain a matrix in reduced row echelon form (RREF). If there are any rows of zeros at the bottom, we remove them. Let the resulting matrix be $B$, with rows $\mathbf{b}_1, \ldots \mathbf{b}_k,\ k \leq m$, and consider the two possible cases:

- **If the rows of $A$ were independent**, there were no rows of zeros, and $k = m$. In this case both $A$ and $B$ correspond to maps from $\mathbb{F}_q^k$ to $\mathbb{F}_q^n$; $f_A(\mathbf{x}) = \mathbf{x}A$ and $f_B(\mathbf{x}) = \mathbf{x}B$. Their images are the same, so we can let

  $$C = \mathrm{im}(f_A) = \mathrm{im}(f_B) = \{\mathbf{x}A \mid \mathbf{x} \in \mathbb{F}_q^k\} = \{\mathbf{x}B \mid \mathbf{x} \in \mathbb{F}_q^k\}.$$

  (But of course in general, for a given $\mathbf{x}$, $\mathbf{x}A \neq \mathbf{x}B$.)

  Moreover, since the rows of $A$ are linearly independent, we know that

  $$(x_1, \ldots, x_k)\begin{pmatrix} - & \mathbf{a}_1 & - \\ & \vdots & \\ - & \mathbf{a}_k & - \end{pmatrix} = x_1\mathbf{a}_1 + \cdots + x_k\mathbf{a}_k = \mathbf{0} \Longrightarrow x_1 = \ldots = x_k = 0.$$

  So$f_A$ is injective, and $\dim(\mathrm{im}(f_A)) = \dim(\mathbb{F}_q^k) = k$. (We know this already as the $\mathbf{a}_i$ are a basis for $\mathrm{im}(f_A)$.) All this is equally true for $B$. Both $A$ and $B$ are generator matrices for $C$.

- **If the rows of $A$ were dependent**, we had to remove at least one row of zeros, and $k < m$. Then for $B$ we can say, as above, that $f_B : \mathbb{F}_q^k \longrightarrow \mathbb{F}_q^n$ is injective, and $\dim(\mathrm{im}(f_B)) = k$. But $f_A : \mathbb{F}_q^m \longrightarrow \mathbb{F}_q^n$ is not injective: We know there are some $\lambda_i$, not all zero, such that $\lambda_1 \mathbf{a}_1 + \cdots + \lambda_m \mathbf{a}_m = \mathbf{0}$. Then if $\mathbf{x} = (\lambda_1, \ldots, \lambda_m)$, we have $\mathbf{x} \neq \mathbf{0}$ but $\mathbf{x}A = \mathbf{0}$. The map $f_A$ maps a larger space onto a smaller, dimension $m$ to dimension $k$. In this case, $B$ is a generator matrix for $C = \mathrm{im}(f_B)$, but A is not.

Thus a generator-matrix not only specifies a code, it also immediately tells us its dimension and so its size.

**Example 25.** Let $C = \langle\{(0,0,3,1,4),(2,4,1,4,0),(5,3,0,1,6)\}\rangle \subseteq F_7^5$. To find $|C|$, and a generator matrix for $C$, we make $A$, and row-reduce:

$$A = \begin{pmatrix} 2 & 4 & 1 & 4 & 0 \\ 5 & 3 & 0 & 1 & 6 \\ 0 & 0 & 3 & 1 & 4 \end{pmatrix} \xrightarrow{A_{12}} \begin{pmatrix} 2 & 4 & 1 & 4 & 0 \\ 0 & 0 & 1 & 5 & 6 \\ 0 & 0 & 3 & 1 & 4 \end{pmatrix} \xrightarrow{M_1(4)} \begin{pmatrix} 1 & 2 & 4 & 2 & 0 \\ 0 & 0 & 1 & 5 & 6 \\ 0 & 0 & 3 & 1 & 4 \end{pmatrix} \xrightarrow{A_{21}(3),A_{23}(4)} \begin{pmatrix} 1 & 2 & 0 & 3 & 4 \\ 0 & 0 & 1 & 5 & 6 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Thus $B = \begin{pmatrix} 1 & 2 & 0 & 3 & 4 \\ 0 & 0 & 1 & 5 & 6 \end{pmatrix}$ is a generator matrix for $C$ , and $|C| = 7^2 = 49$.

$\triangle$

## 3.3 Encoding and Channel Decoding

Recall the basic situation described in Chapter 1. The generator-matrix provides a very easy way to encode messages to codewords: we let our messages be vectors in $\mathbb{F}_q^k$, and then multiply by the generator-matrix $G$ to to obtain codewords in $C \subseteq \mathbb{F}_q^n$.

$$\boxed{\text{SENDER}} \xrightarrow[\mathbf{x}\in\mathbb{F}_q^k]{\text{message}} \boxed{\begin{array}{c}\text{ENCODER}\\ \mathbf{c}=\mathbf{x}G\end{array}} \xrightarrow[\mathbf{c}\in C\subseteq\mathbb{F}_q^n]{\text{codeword}} \boxed{\text{CHANNEL}}$$

It is crucial that the rows of $G$ are independent, so that $f_G$ is injective; otherwise, two different messages would have the same codeword. Also, if the code is to enable us to correct (or even detect) any errors, then there must be some words in $\mathbb{F}_q^n$ but not in $C$, so $k$ must be strictly $< n$. Thus a generator-matrix is always 'landscape' in shape.

The codeword $\mathbf{c}$ then travels through the channel, where it may or may not be changed, and is received as $\mathbf{y} \in \mathbb{F}_q^n$. The receivers do their best to correct any error, finding a codeword $\mathbf{c}'$ which is the (or one of the) most likely to have been sent. (In Chapter 4 we will look again at this process.) And finally, they must find the message corresponding to this codeword, $\mathbf{x}'$ such that $\mathbf{x}'G = \mathbf{c}'$. This last process is called "channel decoding", to distinguish it from the "decoding" which attempts to correct errors. ( In the first pictures in Chapter 1, '$\sim$' is decoding, and '$\to$' is channel decoding.) So at the receiver's end we have:

$$\boxed{\text{CHANNEL}} \xrightarrow[\mathbf{y}\in\mathbb{F}_q^n]{\text{received word}} \boxed{\begin{array}{c}\text{DECODER}\\ \text{error correction?}\end{array}} \xrightarrow[\mathbf{c}'\in C]{\text{codeword}} \boxed{\text{CHANNEL DECODER}} \xrightarrow[\mathbf{x}'\in\mathbb{F}_q^k]{\text{message?}} \boxed{\textit{USER}}$$

But how can we find $\mathbf{x}'$ from $\mathbf{c}'$? Our $G$ is not invertible; it is not even square. Let $\mathbf{x}' = (x'_1, \dots x'_k)$, $\mathbf{c}' = (c'_1, \dots c'_n)$, and $G$ have *columns* $\mathbf{g}_1, \dots \mathbf{g}_n$. Then

$$\mathbf{x}'G = (x_1, \dots, x_k) \begin{pmatrix} | & & | \\ \mathbf{g}_1 & \cdots & \mathbf{g}_n \\ | & & | \end{pmatrix} = (c'_1, \dots, c'_n) = \mathbf{c}'$$

So, using the usual dot product (or scalar product), we have $n$ equations $\mathbf{x}' \cdot \mathbf{g}_i = c'_i$, each in $k$ unknowns, the $x_i$. It is because $\mathbf{c}'$ is a codeword that these equations are consistent and we can find such an $\mathbf{x}'$, and because the map $f_G : \mathbb{F}_q^k \longrightarrow \mathbb{F}_q^n$ is injective that there is only one solution.

You can solve these equations by any method you like; you might prefer to think about $G^t(\mathbf{x}')^t = (\mathbf{c}')^t$, and row-reduce the augmented matrix $(G^t \mid (\mathbf{c}')^t)$. But by picking the right generator-matrix for $C$ in the first place, we can make this much easier. Suppose that $G$ is in RREF. Then $G$ has $k$ columns, $\mathbf{g}_{i_1}, \dots, \mathbf{g}_{i_k}$, which have a leading 1 and zeros elsewhere. Each of these picks out one coordinate of $\mathbf{x}'$, so $c'_{i_j} = \mathbf{x}' \cdot \mathbf{g}_{i_j} = x'_j$. So in this case we can read off $\mathbf{x}'$ from $\mathbf{c}'$ without any calculation.

**Example 26.** Let $C \subseteq \mathbb{F}_7^5$ have generator matrix $G = \begin{pmatrix} 1 & 2 & 0 & 3 & 4 \\ 0 & 0 & 1 & 5 & 6 \end{pmatrix}$, and suppose we have corrected some received word to the codeword (6,5,3,5,0). Then to channel decode, we must find $\mathbf{x}' = (x'_i, x'_2)$ such that

$$(x'_i, x'_2) \begin{pmatrix} 1 & 2 & 0 & 3 & 4 \\ 0 & 0 & 1 & 5 & 6 \end{pmatrix} = (6, 5, 3, 5, 0).$$

Clearly $\mathbf{x}' = (6, 3)$. △

## 3.4 Equivalence and Standard Form

Sometimes two codes have different codewords, but are the same in all the ways that matter.

**Definition 3.2.** Two codes $C_1$, $C_2$ are *equivalent* if we can transform one to the other by applying a sequence of changes of two kinds to all the codewords:

i) permuting the $n$ positions.

ii) in a particular position, permuting the $|A| = q$ symbols.

**Proposition 3.3.** *Two codes which are equivalent have the same parameters* $(n, M, d)$.

**Definition 3.4.** Two linear codes $\subseteq \mathbb{F}_q^n$ are **permutation equivalent** if we can transform one to the other by applying a sequence of permutations to all of the codewords.

**Definition 3.5.** Two linear codes $\subseteq \mathbb{F}_q^n$ are *monomially equivalent* if we can transform one to the other by applying a sequence of permutations to all of the codewords of the following form:

i) permuting the $n$ positions.

ii) in a particular position, multiplying by $\lambda \in \mathbb{F}_q, \lambda \neq 0$ .

Clearly if two codes are permutation equivalent, then they are monomially equivalent, and we will simply never need to use a transformation of type ii) from the definition of monomial equivalence. Also, since $\mathbb{F}_q$ is a field, multiplying by $\lambda \neq 0$ will permute the elements of $\mathbb{F}_q$, but not all permutations of $\mathbb{F}_q$ can be obtained in this way. So if two codes are monomially equivalent, then they are also equivalent, but not vice-versa.

A change of either type is a bijective map $\pi : \mathbb{F}_q^n \longrightarrow \mathbb{F}_q^n$; it just permutes the vectors of $\mathbb{F}_q^n$. Let us write $\Pi$ for a sequence of such changes, so $\Pi = \pi_s \circ \cdots \circ \pi_1$. If there exists such a $\Pi$ with $\Pi(C_1) = C_2$, then $C_1$ and $C_2$ are equivalent, and we write $C_1 \equiv C_2$ if the type of equivalence being considered is clear.

It is also quite easy to see that any such $\pi$ will preserve the linear structure: $\pi(\mathbf{x}+\mathbf{y}) = \pi(\mathbf{x})+\pi(\mathbf{y})$ and $\pi(\lambda\mathbf{x}) = \lambda\pi(\mathbf{x})$, and so the same is true for any $\Pi$. Since $C_1$ and $C_2$ must have the same dimension (as they are both linear codes over the same field with the same number of codewords), then as a consequence of the rank-nullity theorem it follows that $\Pi(C_1) = C_2$ if and only if $\Pi$ changes a basis for $C_1$ into a basis for $C_2$. We can therefore define equivalence in terms of generator-matrices.

**Definition 3.6.** An $m \times m$ matrix $P \in M_{m,m}(\mathbb{Z}_2)$ is a *permutation matrix* if it has a single 1 in each row and column, and zeros elsewhere. Note that any permutation can be written as a permutation matrix of an appropriate size.

Those of you studying Representation Theory III will probably recognise such matrices as those found when considering representations of the symmetric group.

**Proposition 3.7.** *Let $C_1$ and $C_2$ be codes in $\mathbb{F}_q^n$ with generator-matrices $G_1$ and $G_2$ respectively. Then if $G_2 = G_1P$ for some permutation matrix $P$, then $C_1$ and $C_2$ are permutation equivalent.*

**Example 27.** Let $C_1, C_2 \subseteq \mathbb{F}_q^3$, and let $C_1$ have generator matrix $G_1$ where

$$G_1 = \begin{pmatrix} | & | & | \\ \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{g}_3 \\ | & | & | \end{pmatrix}.$$

If

$$G_2 = \begin{pmatrix} | & | & | \\ \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{g}_3 \\ | & | & | \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} | & | & | \\ \mathbf{g}_3 & \mathbf{g}_1 & \mathbf{g}_2 \\ | & | & | \end{pmatrix},$$

is a generator matrix for $C_2$, then by linearity every $c_2 \in C_2$ is the image of a $c_1 \in C_1$ under the permutation given in cycle form as (123). △

**Definition 3.8.** An $m \times m$ matrix $M \in M_{m,m}(\mathbb{F}_q)$ is a *monomial matrix* if it has exactly one non-zero element in each row and column.

A monomial matrix $M$ can always be written as $M = DP$ or $M = PD'$, for $P$ a permutation matrix, and $D, D'$ diagonal matrices. We call $P$ the *permutation part* and $D, D'$ the *diagonal part* of $M$ respectively.

**Example 28.**

$$\underbrace{\begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{D}\underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}}_{P} = \underbrace{\begin{pmatrix} 0 & 2 & 0 \\ 0 & 0 & 3 \\ 1 & 0 & 0 \end{pmatrix}}_{M} = \underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}}_{P}\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}}_{D'}$$

If we apply $M$ to the generator matrix $G_1$ of a code $C_1 \subseteq \mathbb{F}_5^3$, as $G_2 = GM$, then this consists of first multiplying all codewords by 2 in the first position, and by three in the second position, and then by applying the permutation (123). Equivalently, we could apply the permutation first, and then multiply all words by 2 in the second position and by 3 in the third position.

The matrix $G_2$ is then the generator matrix for a monomially equivalent code to $C_1$. △

**Proposition 3.9.** *Let $C_1$ and $C_2$ be codes in $\mathbb{F}_q^n$ with generator-matrices $G_1$ and $G_2$ respectively. Then if $G_2 = G_1M$ for some monomial matrix $M$, then $C_1$ and $C_2$ are monomially equivalent.*

Note that Propositions 3.7 and 3.9 do not say "if and only if". This is because most codes have many possible generator-matrices: if $C_1$ and $C_2$ are linearly equivalent, there must be a $\Pi$ taking $G_1$ to *some* generator-matrix for $C_2$ - but it might not be $G_2$. However, if $C_1$ and $C_2$ are equivalent, and $C_1$ has generator matrix $G_1$, then $G_2 = G_1H$ (for $H$ either a monomial matrix or permutation matrix as appropriate) is a generator matrix for $G_2$.

If we let $S$ be the set of all codes of length $n$ over $\mathbb{F}_q$, then both permutation equivalence and monomial equivalence are *equivalence relations*, meaning that they satisfy the following conditions:

- $C \sim C$, $\forall\, C \in S$  (Reflexive)
- If $C_1 \sim C_2$, then $C_2 \sim C_1$  (Symmetric)
- If $C_1 \sim C_2$ and $C_2 \sim C_3$, then $C_1 \sim C_3$  (Transitive)

We can therefore partition the set of all codes of length $n$ over $\mathbb{F}_q$ into equivalence classes (using either permutation equivalence or monomial equivalence). For those of you who have done Algebra II, this is the *orbit* of the code under the symmetric group.

However, there may be some permutations which don't just send the code to another element of its equivalence class, but rather send the code to itself. In the language of algebra, these are stabilisers of the code, and the set of all stabilisers forms a group under composition.

**Definition 3.10.** The set of permutations sending a code to itself forms a group under composition, with the trivial permutation acting as the identity. This group is known as the *permutation automorphism group* of $C$, and is written as $\mathrm{PAut}(C)$.

In practice, we may think of these permutations either as matrices, or in their cycle form. The two are equivalent as demonstrated in Example 27.

For a code $C$ of block length $n$, $\mathrm{PAut}(C)$ is a subgroup of the symmetric group on $n$ symbols $S_n$, $\mathrm{PAut}(C) \subseteq S_n$.

**Aside:** If you didn't study Algebra II, you may not have seen the definition of a subgroup, so I'll repeat it here.

**Definition.** Given a group $G$, a subset $H \subseteq G$ is said to be a *subgroup* if:

1. The identity $e \in H$,
2. For any $h_1,\ h_2 \in H$, we have $h_1 h_2 \in H$,
3. For any $h \in H$, we have that the inverse $h^{-1} \in H$.

Similarly, the set of all monomial matrices which sends a code to itself also forms a group.

**Definition 3.11.** The set of monomial matrices sending a code to itself forms a group under composition. This group is known as the *monomial automorphism group* of $C$, and is written as $\mathrm{MAut}(C)$.

For a binary code we have that $\mathrm{PAut}(C) = \mathrm{MAut}(C)$, but more generally we have $\mathrm{PAut}(C) \subseteq \mathrm{MAut}(C)$.

The idea of equivalence allows us to work with particularly convenient generator-matrices:

**Definition 3.12.** For $k \leq n$, if a $k \times n$ generator-matrix is of the form $(I_k \mid A)$, we say that it is in **standard form**.

Here $I_k$ is the $k \times k$ identity matrix, and $A$ some $k \times (n-k)$ matrix.

**Proposition 3.13.** *Any linear code is permutation equivalent to one with a generator-matrix in standard form.*

*Proof.* For any code $C_1 \subseteq \mathbb{F}_q^n$ we can find a generator-matrix $G_1$ which is in RREF. Then by permuting columns we can obtain $G_2$ of form $(I \mid A)$, and this is the generator-matrix of an equivalent code $C_2$. □

One advantage of standard form is that it makes channel decoding even easier. The first $k$ digits of the codeword are the corresponding message.

**Example 29.** Let $C_1 \subseteq \mathbb{F}_7^5$ have generator matrix $G_1 = \begin{pmatrix} 1 & 2 & 0 & 3 & 4 \\ 0 & 0 & 1 & 5 & 6 \end{pmatrix}$ as above. By swapping columns 2 and 3, we obtain $G_2 = \begin{pmatrix} 1 & 0 & 2 & 3 & 4 \\ 0 & 1 & 0 & 5 & 6 \end{pmatrix}$, which is in standard form. Then using the equivalent code $C_2$ which has this generator matrix, a sender would encode message (1,5) as

$$(1,5)\begin{pmatrix} 1 & 0 & 2 & 3 & 4 \\ 0 & 1 & 0 & 5 & 6 \end{pmatrix} = (1,5,2,0,6),$$

and a receiver would channel decode (6,3,5,5,0) to (6,3). △

To end this chapter I will briefly introduce some terminology which we will not stress, but you may meet in other sources.

Suppose we encode using a generator-matrix in standard form. Then the first $k$ digits of the codeword are the message, and the remaining $n - k$ digits, calculated from the message using the $A$ part of the generator-matrix, are sometimes called **check-digits**. Their role is to show up symbol-errors in the message digits: if only a message-digit is changed, the result will not be a codeword. Syndrome decoding (Section 4.3) exploits this kind of idea.

The **rank** of a code is its dimension, $k$. (This is also the rank, in the linear algebra sense, of its generator-matrix.) Its **redundancy** is $n - k$, the number of check-digits. These are "redundant" in the sense that they are not the message, though they are certainly not useless. And the **rate** of the code is $k/n$, the fraction of the stream of symbols which is the actual messages. Redundancy and rate are most easily explained for a code with generator-matrix in standard form, but these names can be used for the parameters $n-k$ and $k/n$ of any linear code.