ENEE 420
FALL 2012
COMMUNICATIONS SYSTEMS

ERROR CONTROL AND CODING: GENERAL FACTS

Coding Theory is at the intersection of Algebra, Geometry and Probability Theory.

Hamming spaces

We begin with some basic notions concerning Hamming spaces.

With positive integer p, we define the Hamming space $\mathcal{H}_p$ of dimension p as the vector space $\{0,1\}^p$ equipped with modulo-2 addition and multiplication. More specifically, with $\boldsymbol{x} = (x_1, \ldots, x_p)$ and $\boldsymbol{y} = (y_1, \ldots, y_p)$ in $\mathcal{H}_p$, we define their addition as the vector $\boldsymbol{x} + \boldsymbol{y}$ (mod 2) defined componentwise by

$$(\boldsymbol{x} + \boldsymbol{y})_\ell = (x_\ell + y_\ell) \pmod 2, \quad \ell = 1, \ldots, p.$$

Furthermore, we have

$$t\boldsymbol{x} = (tx_1, \ldots, tx_p). \quad t = 0, 1.$$

The *Hamming distance* on $\mathcal{H}_p$ is the mapping $\mathcal{H}_p \times \mathcal{H}_p \to \mathbb{R}_+$ defined by

$$d_H(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\ell=1}^{p} |x_\ell - y_\ell|, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}_p.$$

Thus, $d_H(\boldsymbol{x}, \boldsymbol{y})$ counts the number of positions where the vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ differ. That the mapping $\mathcal{H}_p \times \mathcal{H}_p \to \mathbb{R}_+$ is a *distance* on $\mathcal{H}^p$ can be easily established. Indeed, it satisfies the properties of definiteness and symmetry, and the triangular inequality:

- Definiteness:
$$d_H(\boldsymbol{x}, \boldsymbol{y}) = 0 \quad \text{if and only if} \quad \boldsymbol{x} = \boldsymbol{y}.$$
- Symmetry: For all $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathcal{H}_p$, we have
$$d_H(\boldsymbol{y}, \boldsymbol{x}) = d_H(\boldsymbol{x}, \boldsymbol{y})$$

- Triangular inequality: For all $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ in $\mathcal{H}_p$, we have

$$d_H(\boldsymbol{x}, \boldsymbol{z}) \leq d_H(\boldsymbol{x}, \boldsymbol{y}) + d_H(\boldsymbol{y}, \boldsymbol{z}).$$

We set

$$w_H(\boldsymbol{x}) = \sum_{\ell=1}^{p} x_\ell, \quad \boldsymbol{x} \in \mathcal{H}_p.$$

The quantity $w_H(\boldsymbol{x})$ is called the *(Hamming) weight* of the element $\boldsymbol{x}$. It counts the number of coordinates of $\boldsymbol{x}$ which are non-zero. By direct inspection we note that

$$d_H(\boldsymbol{x}, \boldsymbol{y}) = w_H(\boldsymbol{x} - \boldsymbol{y}), \quad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}_p. \tag{1}$$

Generalities on codes

Throughout k and n are positive inetgers such that $k < n$. The information (message) to be transmitted is formatted as a vector $\boldsymbol{m}$ in $\mathcal{H}_k$ with binary components, say

$$\boldsymbol{m} = (m_1, \ldots, m_k).$$

An (n, k)-*code* is any *deterministic* mapping $C : \mathcal{H}_k \to \mathcal{H}_n$ with the interpretation that the *codeword* $C(\boldsymbol{m})$ is *uniquely* associated with the information vector $\boldsymbol{m}$. This requires the mapping $C : \mathcal{H}_k \to \mathcal{H}_n$ to be *one-to-one*, i.e., for distinct $\boldsymbol{m}$ and $\boldsymbol{m}'$ in $\mathcal{H}_k$, we have $C(\boldsymbol{m}) \neq C(\boldsymbol{m}')$. Conversely, if $C(\boldsymbol{m}) = C(\boldsymbol{m}')$, then necessarily $\boldsymbol{m} = \boldsymbol{m}'$. On occasions we shall write

$$\boldsymbol{c} = (c_1, \ldots, c_n) = C(\boldsymbol{m}).$$

The process by which $\boldsymbol{c}$ is associated with $\boldsymbol{m}$ is known as *encoding*, and the mapping $C : \mathcal{H}_k \to \mathcal{H}_n$ is also referred to as an encoding mapping.

The collection

$$\mathcal{C} = \{\boldsymbol{m} \in \mathcal{H}_k : \ C(\boldsymbol{m})\}$$

is known as the *codebook* associated with the code $C : \mathcal{H}_k \to \mathcal{H}_n$, and in many instances it is customary to refer to $\mathcal{C}$ as the code (without any further reference to the encoding mapping $C : \mathcal{H}_k \to \mathcal{H}_n$). Since the mapping $C : \mathcal{H}_k \to \mathcal{H}_n$ is one-to-one, we have $|\mathcal{C}| = |\mathcal{H}_k|$, whence

$$|\mathcal{C}| = 2^k$$

upon recalling that $|\mathcal{H}_k| = 2^k$. It should be pointed out that distinct encoding mappings can generate the same codebook.

A code $C : \mathcal{H}_k \to \mathcal{H}_n$ is said to be in *systematic* form if

$$\boldsymbol{c} = C(\boldsymbol{m}) = (\boldsymbol{b}, \boldsymbol{m}), \quad \boldsymbol{m} \in \mathcal{H}_k$$

where $\boldsymbol{b}$ is an element of $\mathcal{H}_{n-k}$ determined by $\boldsymbol{m}$. The advantage of a systematic code lies in the fact that $\boldsymbol{m}$ can be read off $\boldsymbol{c}$ without any further processing.

The *minimum (Hamming) distance* of the code C is simply defined as

$$d_H(C) := \min \left(d_H(C(\boldsymbol{m}), C(\boldsymbol{m}')), \quad \begin{array}{c} \boldsymbol{m} \neq \boldsymbol{m}' \\ \boldsymbol{m}, \boldsymbol{m}' \in \mathcal{H}_k \end{array} \right).$$

It is plain that $d_H(C)$ is always a *positive* integer. This quantity measures how *clustered* the codewords in $\mathcal{C}$ are in $\mathcal{H}_n$; its importance will shortly become apparent. Moreover, note that $d_H(C)$ depends only on the codebook generated by the encoding mapping $C : \mathcal{H}_k \to \mathcal{H}_n$, and not on the specific encoding mapping used. Sometimes we shall also use the notation $d_H(\mathcal{C})$ to reflect this fact.

The decoding problem

A source generates information that needs to be transmitted over an imperfect channel: Possibly after some source coding, the information (message) to be transmitted is formatted as an information vector $\boldsymbol{m}$ in $\mathcal{H}_k$. Using a (n, k)-code $C : \mathcal{H}_k \to \mathcal{H}_n$, this message is encoded into a codeword $\boldsymbol{c} = C(\boldsymbol{m})$ in $\mathcal{H}_n$. This codeword is fed into the modulator which then generates the appropriate waveform used for transmission over the channel. The transmission process being imperfect, impairments occur due to channel noise and channel distortions (e.g., dispersive effects, fading, etc). As a result, the received waveform may be different from the modulated waveform that was originally sent over the channel. The demodulation process, which is expected to invert the modulation process, then extracts (from the received waveform) a "received" vector $\boldsymbol{r}$ in $\mathcal{H}_n$.

Under reasonable channel conditions, we expect $\boldsymbol{r}$ to be a reasonably good proxy for $\boldsymbol{c}$. However, there is no guarantee that $\boldsymbol{r} = \boldsymbol{c}$, or even that $\boldsymbol{r}$ is a codeword in $\mathcal{C}$. The need therefore arises to reverse the encoding process by providing a guess or estimate of $\boldsymbol{c}$ on the basis of $\boldsymbol{r}$. This process is known as the *decoding* process, and can be formalized through a mapping $D : \mathcal{H}_n \to \mathcal{C}$ known as a *decoding* mapping. Once a decoding mapping has been selected, the codeword $\widehat{\boldsymbol{c}} = D(\boldsymbol{r})$ is determined, and the unique message $\widehat{\boldsymbol{m}}$ such that

$$\widehat{\boldsymbol{c}} = C(\widehat{\boldsymbol{m}})$$

can be identified. It is then concluded that the message $\widehat{\boldsymbol{m}}$ has been sent. Of course, if

$$\widehat{\boldsymbol{c}} = D(\boldsymbol{r}) \neq \boldsymbol{c},$$

then there will be an error since $\widehat{\boldsymbol{m}} \neq \boldsymbol{m}$!

There are many ways to design decoding mappings. However, any such selection must have the following property: If the received vector $\boldsymbol{r}$ is already a codeword, then the decoding mapping $D : \mathcal{H}_n \to \mathcal{C}$ should return $\boldsymbol{r}$, i.e.,

$$\widehat{\boldsymbol{c}} = D(\boldsymbol{r}) = \boldsymbol{r} \quad \text{if } \boldsymbol{r} \in \mathcal{C}. \tag{2}$$

See below for the rational for doing so.

Upon reception of $\boldsymbol{r}$, two possibilities arise:

1. If $\boldsymbol{r} \in \mathcal{C}$: Since $\boldsymbol{r}$ is already a codeword, there is no reason to declare that an error has occured. The receiving side, having no evidence to the contrary, will indeed accept the received vector $\boldsymbol{r}$ as correct – This is the reason why any decoding mapping $D : \mathcal{H}_n \to \mathcal{C}$ satisfies (2). More precisely, the receiver will accept $\boldsymbol{r}$ as having been the codeword that was used as input to the modulation process. Consequently, since $\boldsymbol{r}$ is a *bona fide* codeword, there exists a unique element $\widehat{\boldsymbol{m}}$ in $\mathcal{H}_k$ so that

   $$\boldsymbol{r} = C(\widehat{\boldsymbol{m}}),$$

   and it will be concluded that $\widehat{\boldsymbol{m}}$ was indeed the information transmitted! However there is no guarantee that $\boldsymbol{r} = \boldsymbol{c}$. It is possible that $\boldsymbol{r} \neq \boldsymbol{c}$, still with $\boldsymbol{r}$ an element of $\mathcal{C}$, in which case a decoding error occurs. Such an error *cannot* be detected. Such a failure to detect an error is characterized by

   $$\boldsymbol{r} \in \mathcal{C} \quad \text{but} \quad \boldsymbol{r} \neq \boldsymbol{c}.$$

2. If $\boldsymbol{r} \notin \mathcal{C}$: Error detection occurs since $\boldsymbol{r}$ is *not* a codeword. Error *correction* is then required, and this amounts to selecting an *estimate* $\widehat{\boldsymbol{c}}$ of $\boldsymbol{c}$ on the basis $\boldsymbol{r}$. This estimate $\widehat{\boldsymbol{c}}$ must be a codeword and is produced by the decoding mapping $D : \mathcal{H}_n \to \mathcal{C}$. Once the estimate $\widehat{\boldsymbol{c}} = D(\boldsymbol{r})$ has been computed, there exists a unique element $\widehat{\boldsymbol{m}}$ in $\mathcal{H}_k$ so that

   $$\widehat{\boldsymbol{c}} = C(\widehat{\boldsymbol{m}}).$$

   It will then be concluded that $\widehat{\boldsymbol{m}}$ was indeed the information transmitted! This conclusion may be in error since there is a possibility that $\widehat{\boldsymbol{c}} \neq \boldsymbol{c}$. Of course, if the estimation algorithm returns $\widehat{\boldsymbol{c}} = \boldsymbol{c}$, then $\widehat{\boldsymbol{m}} = \boldsymbol{m}$, in which case both error detection and error correction would have taken place!

One very popular algorithm, known as Nearest Neighbor decoding, is now described. The Nearest Neighbor decoding rule selects an estimate $\widehat{\boldsymbol{c}}_{\rm Near}$ of $\boldsymbol{c}$ on the basis of $\boldsymbol{r}$ through the rule

$$\widehat{\boldsymbol{c}}_{\rm Near} := \arg\min\left(\boldsymbol{c}_{\rm other} \in \mathcal{C} : \ d_H(\boldsymbol{c}_{\rm other}, \boldsymbol{r})\right)$$

with a tie-breaker. The geometric interpretation of this rule is clear: The estimate $\widehat{\boldsymbol{c}}_{\rm Near}$ is the codeword in $\mathcal{C}$ that is closest (in the sense of Hamming distance) to the received $\boldsymbol{r}$. This rule satisfies both requirements mentioned above.

Basic facts

Consider a (n,k)-code $C : \mathcal{H}_k \to \mathcal{H}_n$. The minimum Hamming distance of the code C is a useful measure of its ability to detect and correct errors. This will become apparent from the following discussion: Throughout the message $\boldsymbol{m}$ in $\mathcal{H}_k$ is encoded with the codeword $\boldsymbol{c} = C(\boldsymbol{m})$, and the received vector $\boldsymbol{r}$ is of the form

$$\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e} \pmod 2 \tag{3}$$

where the vector $\boldsymbol{e}$ is an element of $\mathcal{H}_n$ which describes the channel errors.

Lemma 0.1 *Error detection: Consider a (n,k)-code $C : \mathcal{H}_k \to \mathcal{H}_n$ with*

$$d_H(\mathcal{C}) = d + 1$$

for some positive integer d. This code has the ability to detect any error pattern with k bit reversals provided $1 \le k \le d$.

In other words, any pattern with *at most* d errors, if it occurs, will be detected. This property does not depend on the decoding scheme used when the received vector $\boldsymbol{r}$ is not a codeword.

Proof. The message $\boldsymbol{m}$ in $\mathcal{H}_k$ is encoded with the codeword $\boldsymbol{c} = C(\boldsymbol{m})$, and the received vector $\boldsymbol{r}$ is of the form (3) for some error vector $\boldsymbol{e}$ in $\mathcal{H}_n$.

Assume that $\boldsymbol{r}$ differs from $\boldsymbol{c}$ in exactly k positions with

$$1 \le k < d_H(\mathcal{C}).$$

Obviously, since $d_H(\boldsymbol{r}, \boldsymbol{c}) = k$, we conclude that

$$0 < d_H(\boldsymbol{r}, \boldsymbol{c}) < d_H(\mathcal{C}).$$

As a result, $\boldsymbol{r}$, which is different from $\boldsymbol{c}$, cannot be a codeword in $\mathcal{C}$ and the error is detected ■

Lemma 0.2 *Error correction: Consider a (n,k)-code $C : \mathcal{H}_k \to \mathcal{H}_n$ with*

$$d_H(\mathcal{C}) = 2d + 1$$

for some positive integer d. This code has the ability to correct any error pattern with k bit reversals provided $1 \le k \le d$ provided Nearest Neigbor decoding is used.

Any pattern with *at most* d errors will be detected and subsequently corrected

Proof. Again, the message $\boldsymbol{m}$ in $\mathcal{H}_k$ is encoded with the codeword $\boldsymbol{c} = C(\boldsymbol{m})$, and the received vector $\boldsymbol{r}$ is of the form (3) for some error vector $\boldsymbol{e}$ in $\mathcal{H}_n$.

Assume that $\boldsymbol{r}$ differs from $\boldsymbol{c}$ in exactly k positions with

$$0 < 2k < d_H(\mathcal{C}).$$

We already have $k < d_H(\mathcal{C})$ and that error pattern will be detected by Lemma 0.1.

By the triangle inequality we have

$$d_H(\boldsymbol{c}, \boldsymbol{c}_{\rm other}) \le d_H(\boldsymbol{c}, \boldsymbol{r}) + d_H(\boldsymbol{r}, \boldsymbol{c}_{\rm other}), \quad \boldsymbol{c}_{\rm other} \in \mathcal{C}.$$

Pick any other codeword $\boldsymbol{c}_{\rm other} \neq \boldsymbol{c}$ in $\mathcal{C}$. Since $d_H(\boldsymbol{c}, \boldsymbol{r}) = k$, we have

$$d_H(\boldsymbol{c}, \boldsymbol{c}_{\rm other}) \le k + d_H(\boldsymbol{r}, \boldsymbol{c}_{\rm other}).$$

Using the definition of $d_H(\mathcal{C})$ yields

$$d_H(\mathcal{C}) \le k + d_H(\boldsymbol{r}, \boldsymbol{c}_{\rm other}),$$

whence

$$d_H(\mathcal{C}) - k \le d_H(\boldsymbol{r}, \boldsymbol{c}_{\rm other}).$$

By the condition on $d_H(\mathcal{C})$, we conclude that

$$d_H(\boldsymbol{r}, \boldsymbol{c}_{\rm other}) > k, \qquad \begin{matrix} \boldsymbol{c}_{\rm other} \neq \boldsymbol{c} \\ \boldsymbol{c}_{\rm other} \in \mathcal{C}. \end{matrix}$$

while

$$d_H(\boldsymbol{r}, \boldsymbol{c}) = k.$$

Nearest Neighbor decoding yields

$$\begin{aligned} \widehat{\boldsymbol{c}}_{\rm Near} &= \arg\min\left(\boldsymbol{c}_{\rm other} \in \mathcal{C} : \ d_H(\boldsymbol{r}, \boldsymbol{c}_{\rm other})\right) \\ &= \boldsymbol{c}, \end{aligned}$$

and error correction takes place! ■

Channel models

At some abstract level, a communication channel can be viewed as an operation that maps codewords into received vectors. Owing to the vagaries of the communication process, this mapping is usually *random* in that if the codeword $\boldsymbol{c}$ is sent, then only the likelihood of occurence of the received vector $\boldsymbol{r}$ can be specified (if at all). There is no guarantee that the received vector will coincide with $\boldsymbol{c}$, let alone that the identity of $\boldsymbol{r}$ will be known in advance. This state of affairs suggests the following probabilistic characterization of a channel:

A channel is characterized by a collection

$$\mathcal{P} \equiv \{p(\boldsymbol{x}, \boldsymbol{y}), \ \boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}_n\} \tag{4}$$

of scalars such that

$$0 \le p(\boldsymbol{x}, \boldsymbol{y}) \le 1, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}_n$$

and

$$\sum_{\boldsymbol{y} \in \mathcal{H}_n} p(\boldsymbol{x}, \boldsymbol{y}) = 1, \quad \boldsymbol{x} \in \mathcal{H}_n.$$

Thus, for each $\boldsymbol{x}$ in $\mathcal{H}_n$, the collection $\{p(\boldsymbol{x}, \boldsymbol{y}), \ \boldsymbol{y} \in \mathcal{H}_n\}$ is a probability mass function on $\mathcal{H}_n$. We understand $p(\boldsymbol{x}, \boldsymbol{y})$ as the *conditional* probability that $\boldsymbol{y}$ was received given that $\boldsymbol{x}$ was sent, i.e.,

$$p(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{P}\left[\ \boldsymbol{y} \text{ received} | \ \boldsymbol{x} \text{ sent}\right], \quad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}_n.$$

In many cases the channel is *translation invariant* in that there exists a mapping $q : \mathcal{H}_n \to [0, 1]$ such that

$$p(\boldsymbol{x}, \boldsymbol{y}) = q(\boldsymbol{y} - \boldsymbol{x}) \tag{5}$$

$$= q(\boldsymbol{e}), \quad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}_n \tag{6}$$

where $\boldsymbol{e}$ is the channel error determined by

$$\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{x} \pmod 2,$$

or equivalently,

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{e} \pmod 2.$$

The following two models will be used to illustrate a number of issues.

The vector error model – All channel error vectors are assumed to be *equally likely*, i.e.,

$$p(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2^n}, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}_n.$$

This model is clearly translation invariant with

$$q(\boldsymbol{e}) = \frac{1}{2^n}, \quad \boldsymbol{e} \in \mathcal{H}_n.$$

All errors equally matter!

The componentwise model This model is sometimes also known as the *random bit error* model. Bit errors occur *independently* of each other with probability α ($0 < \alpha < 1$), so that

$$\begin{aligned} p(\boldsymbol{x}, \boldsymbol{y}) &= \alpha^{d_H(\boldsymbol{x},\boldsymbol{y})}(1-\alpha)^{n-d_H(\boldsymbol{x},\boldsymbol{y})} \\ &= \alpha^{w_H(\boldsymbol{y}-\boldsymbol{x})}(1-\alpha)^{n-w_H(\boldsymbol{y}-\boldsymbol{x})}, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}_n \end{aligned}$$

as we note that $d_H(\boldsymbol{x}, \boldsymbol{y}) = w_H(\boldsymbol{y} - \boldsymbol{x})$. Thus, this model is also translation invariant with

$$q(\boldsymbol{e}) = \alpha^{w_H(\boldsymbol{e})}(1-\alpha)^{n-w_H(\boldsymbol{e})}, \quad \boldsymbol{e} \in \mathcal{H}_n.$$

For $0 < \alpha < 0.5$, we note that

$$q(\boldsymbol{e}) \downarrow \quad \text{as} \quad w_H(\boldsymbol{e}) \uparrow .$$

so that only errors in a few positions matter!

Performance metrics

Consider a code $C : \mathcal{H}_k \rightarrow\!\!\!\!\rightarrow \mathcal{H}_n$ with codebook $\mathcal{C}$, and assume that the communication channel is characterized by the collection $\mathcal{P}$ given by (4). We are interested in assessing the probability $\mathrm{Err}(C)$ that *error detection fails* when using C. More precisely, assume the message $\boldsymbol{m}$ in $\mathcal{H}_k$ is encoded with the codeword $\boldsymbol{c} = C(\boldsymbol{m})$, and the received vector $\boldsymbol{r}$ is of the form (3). Failure to detect an error is characterized by

$$\boldsymbol{r} \in \mathcal{C} \quad \text{but} \quad \boldsymbol{r} \neq \boldsymbol{c}.$$

In terms of the error pattern $\boldsymbol{e}$ appearing in (3), this is equivalent to

$$\boldsymbol{c} + \boldsymbol{e} \in \mathcal{C} \quad \text{but} \quad \boldsymbol{e} \neq \mathbf{0}_n.$$

Thus, by the law of total probabilities, we get

$$\begin{aligned}
\mathrm{Err}(C) &= \mathbb{P}\left[\text{ Error detection fails under } C\right] \\
&= \sum_{c \in \mathcal{C}} \mathbb{P}\left[\text{ Error detection fails under } C | \boldsymbol{c} \text{ sent }\right] \mathbb{P}\left[\boldsymbol{c} \text{ sent }\right] \\
&= \sum_{c \in \mathcal{C}} \mathbb{P}\left[\boldsymbol{r} \in \mathcal{C},\ \boldsymbol{r} \neq \boldsymbol{c} | \boldsymbol{c} \text{ sent }\right] \mathbb{P}\left[\boldsymbol{c} \text{ sent }\right]. \qquad (7)
\end{aligned}$$

Therefore, as we define

$$\mathcal{E}(\boldsymbol{c}) = \left\{ \boldsymbol{e} \in \mathcal{H}_n :\ \boldsymbol{e} \neq \mathbf{0}_n,\ \boldsymbol{c} + \boldsymbol{e} \in \mathcal{C} \right\},$$

we get

$$\begin{aligned}
\mathbb{P}\left[\boldsymbol{r} \in \mathcal{C}, \boldsymbol{r} \neq \boldsymbol{c} | \boldsymbol{c} \text{ sent }\right] &= \sum_{\boldsymbol{e} \in \mathcal{H}_n} \mathbb{P}\left[\boldsymbol{c} + \boldsymbol{e} \in \mathcal{C},\ \boldsymbol{e} \neq \mathbf{0}_n | \boldsymbol{c} \text{ sent }\right] \\
&= \sum_{\boldsymbol{e} \in \mathcal{E}(\boldsymbol{c})} \mathbb{P}\left[\boldsymbol{c} + \boldsymbol{e} \in \mathcal{C}, \boldsymbol{e} \neq \mathbf{0}_n | \boldsymbol{c} \text{ sent }\right] \\
&= \sum_{\boldsymbol{e} \in \mathcal{E}(\boldsymbol{c})} p(\boldsymbol{c}, \boldsymbol{c} + \boldsymbol{e}). \qquad (8)
\end{aligned}$$

Substituting (8) into (7) we conclude that

$$\begin{aligned}
\mathrm{Err}(C) &= \mathbb{P}\left[\text{ Error detection fails under } C\right] \\
&= \sum_{c \in \mathcal{C}} \left(\sum_{\boldsymbol{e} \in \mathcal{E}(\boldsymbol{c})} p(\boldsymbol{c}, \boldsymbol{c} + \boldsymbol{e}) \right) \mathbb{P}\left[\boldsymbol{c} \text{ sent }\right]. \qquad (9)
\end{aligned}$$

When the channel is translation invariant, this last expression can be rewritten as

$$\mathrm{Err}(C) = \sum_{c\in\mathcal{C}} \left(\sum_{\boldsymbol{e}\in\mathcal{E}(\boldsymbol{c})} q(\boldsymbol{e}) \right) \mathbb{P}\left[\boldsymbol{c} \text{ sent }\right]. \tag{10}$$

In a number of important cases, the code has the property that the sets $\mathcal{E}(\boldsymbol{c})$ are all the same regardless of the choice of $\boldsymbol{c}$, i.e.,

$$\mathcal{E}(\boldsymbol{c}) = \mathcal{E}(\mathbf{0}_n), \quad c \in \mathcal{C}. \tag{11}$$

Lemma 0.3 *Consider a code $C : \mathcal{H}_k \rightarrow\!\!\!\!\rightarrow \mathcal{H}_n$ which satisfies the message invariance property (11), while the communication channel $\mathcal{P}$ satisfies the invariance property (6). Then, we have*

$$\mathrm{Err}(C) = \left(\sum_{\boldsymbol{e}\in\mathcal{E}(\mathbf{0}_n)} q(\boldsymbol{e}) \right). \tag{12}$$

This expression is independent of the pmf $\{\mathbb{P}\left[\boldsymbol{c} \text{ sent }\right],\ \boldsymbol{c}\in\mathcal{C}\}$ on the channel input.

Proof. With the notation used thus far, under message invariance property (11), the expression (10) becomes

$$\begin{aligned} \mathrm{Err}(C) &= \sum_{c\in\mathcal{C}} \left(\sum_{\boldsymbol{e}\in\mathcal{E}(\mathbf{0}_n)} q(\boldsymbol{e}) \right) \mathbb{P}\left[\boldsymbol{c} \text{ sent }\right] \\ &= \left(\sum_{\boldsymbol{e}\in\mathcal{E}(\mathbf{0}_n)} q(\boldsymbol{e}) \right) \left(\sum_{c\in\mathcal{C}} \mathbb{P}\left[\boldsymbol{c} \text{ sent }\right] \right) \end{aligned} \tag{13}$$

and the conclusion (12) follows since

$$\sum_{c\in\mathcal{C}} \mathbb{P}\left[\boldsymbol{c} \text{ sent }\right] = 1.$$

■