A Reed–Solomon Code Magic Trick

TODD D. MATEER
Howard Community College
Columbia, Maryland 21044
tmateer@howardcc.edu

Richard Ehrenborg [**1**] has provided a nice magic trick that can be used to illustrate many properties of Hamming codes. His paper includes a set of manipulatives that can be used to implement the trick. An improved version of this Hamming code magic trick is described in a recent paper in *Math Horizons* [**3**].

In this paper, we introduce a similar magic trick that is based on a Reed–Solomon code. It can be used to introduce Reed–Solomon codes to engineering students, and yet is simple enough to use with high school students. The ideas behind the magic trick are also the basis for the error correction technique used on CDs and DVDs.

The magic trick

The magic trick uses the four cards shown in FIGURE 1.

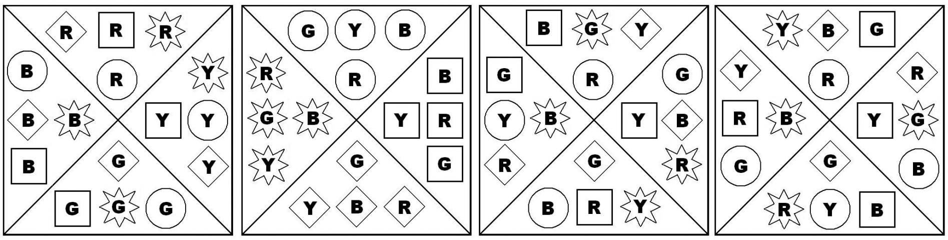
R R R B R Y B B Y Y B G Y G G G Y
G Y B R R B G B Y R Y G G Y B R
B G Y G R G Y B Y B R G R B R Y
Y B G Y R R R B Y G G G B R Y B

Figure 1 Cards used for magic trick

On each card are sixteen different symbols: four shapes, each in four different colors. In this paper, we use the colors Red (R), Yellow (Y), Green (G), and Blue (B) along with the shapes Circle, Square, Diamond, and Star. Any colors and any shapes with the proper symmetry would serve as well. We have put the first letter of the color in the center of each symbol as an extra cue.

A volunteer is selected from the audience and asked to silently pick a color from {Red, Yellow, Blue, Green} and a shape from {Circle, Star, Diamond, Square}. The magician shows the volunteer the four cards, and instructs the volunteer to report the quadrant (Up, Left, Right, Down) where the chosen symbol is found on each card. However, the volunteer is also instructed to lie—that is, give an incorrect response—on one of the cards. Following each response, the magician inconspicuously rotates the card so that the chosen section is in the Up position. As soon as the volunteer responds to the fourth card, the magician tells the volunteer which symbol was selected as well as which card was reported incorrectly.

The secret

The cards are designed so that when they are rotated according to the above instructions, the chosen symbol appears in the same position within the upper section on three of the four cards. None of the remaining symbols will have this property. The magician can then easily tell which symbol was chosen. For the card on which the volunteer lied, the chosen symbol does not appear in its proper position, allowing the magician to recognize the card that was reported incorrectly.

For example, suppose that the volunteer responded "Up Down Up Right" to the cards from FIGURE 1. After the four cards have been rotated, they will appear as shown in FIGURE 2. The magician sees that the Red Diamond is in the upper left position in the top quadrant of the first, second, and fourth cards. So the magician declares that the chosen symbol was the Red Diamond. Since this symbol does not appear in this position on the third card, the magician declares that the third card was reported incorrectly.

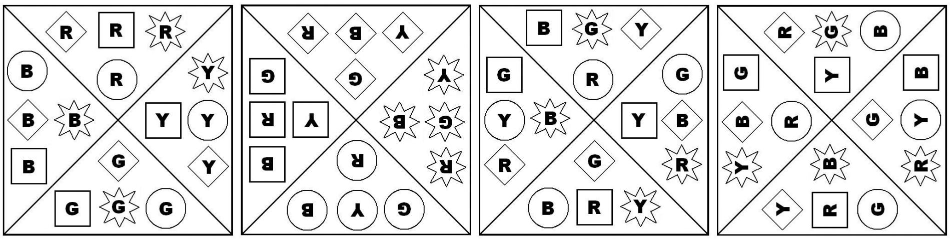

Figure 2 Example of the magic trick

There is nothing special about the Up position in this trick; the magician can rotate the cards in some other direction if desired for variety.

Sometimes, the magician can identify the chosen symbol after the third card and also knows that the volunteer will lie on the fourth card. In this case, the magician can wait for the fourth response or can say "You are about to lie to me, aren't you?" before the fourth card is shown.

The trick works no matter which symbol is chosen and which card is reported incorrectly. The magic is based on properties of a particular Reed–Solomon code. In the remainder of this paper we introduce the Reed–Solomon code and explain why the trick works.

Error-correcting codes

The magic trick is based on one of the simplest Reed–Solomon codes.

On the first two cards, the volunteer can choose any of four possible sections (Up, Left, Right, Down). If the person does not lie on the first two cards, the responses to these two cards uniquely determine the chosen symbol. Let us assign the symbol 0 to represent the upper quadrant, 1 to represent the right quadrant, A to represent the lower quadrant, and B to represent the left quadrant. (These choices will be important later.) We can now represent each symbol using a *codeword* determined by the location of the symbol on the first two cards. For example, if the cards are presented according to FIGURE 1, then the red diamond would be represented using the codeword $[0, A]$.

Suppose that the third card is added and the person responds honestly on each of the three cards. Our code could be modified to use all three replies as given in FIGURE 3. Each three-component vector is now a codeword of the new three-letter code. While each codeword has three letters, only the first two letters are really needed to represent each symbol.

Symbol	Codeword	Symbol	Codeword
R○	UUU = [0 , 0 , 0]	G○	DUR = [A , 0 , 1]
R■	URD = [0 , 1 , A]	G■	DRL = [A , 1 , B]
R◇	UDL = [0 , A , B]	G◇	DDD = [A , A , A]
R★	ULR = [0 , B , 1]	G★	DLU = [A , B , 0]
Y○	RUL = [1 , 0 , B]	B○	LUD = [B , 0 , A]
Y■	RRR = [1 , 1 , 1]	B■	LRU = [B , 1 , 0]
Y◇	RDU = [1 , A , 0]	B◇	LDR = [B , A , 1]
Y★	RLD = [1 , B , A]	B★	LLL = [B , B , B]

Figure 3 A $(3, 2, 2)$ Reed–Solomon code based on the magic trick

In general, a code such as this one can be represented using the two parameters (n, k). Here, n is the total number of letters in each codeword and k is the number of *information components*, that is, the number of letters actually needed to identify the symbol. In the case of the magic trick, the codewords in FIGURE 3 represent a $(3, 2)$ code, since each codeword consists of 3 letters and the first 2 of these components identify the chosen symbol (the color and the shape). A popular set of "real-world" Reed–Solomon code parameters is $(255, 223)$, where 223 components of information are encoded into each 255-letter codeword.

During transmission, a codeword can become distorted. In the case of the magic trick, the codeword is distorted by the one lie of the volunteer. In the case of the CD or DVD, the distortions are in the form of scratches or fingerprints. The extra $n - k$ components added to the k components of information can be used to correct a limited number of these mistakes. For this reason, these codes are called *error-correcting codes*. It is the job of a decoder to determine which codeword best represents the received set of letters after transmission and possible distortion. The best representation is typically the codeword that differs from what was received in the minimum number of positions.

If we study the codewords in FIGURE 3, we see that each codeword differs from any other codeword in at least 2 positions. We call this the *minimum distance* of the code and use the parameter d to represent it. The minimum distance is frequently given as a third parameter of an error-correcting code. So the code of FIGURE 3 can be called a $(3, 2, 2)$ code. It can be shown [**4**] that a code with minimum distance d can correct up to $\lfloor (d - 1)/2 \rfloor$ errors. In the case of the magic trick, this tells us that the $(3, 2, 2)$ code constructed using the first three cards cannot correct any errors.

To see this, suppose that the volunteer lied on one of the first three cards. For example, suppose that the responses of the volunteer were "Left, Up, Right". Using the quadrant symbol assignments mentioned earlier, these responses can be represented as $[B, 0, 1]$. Observe in FIGURE 3 that there are three codewords that differ from $[B, 0, 1]$ in one position ($[A, 0, 1]$, $[B, 0, A]$, and $[B, A, 1]$). Since each of

$\{[A, 0, 1], [B, 0, A], [B, A, 1]\}$ differs from any of the other two codewords in this set in exactly two positions, the vector $[B, 0, 1]$ can be considered to be exactly halfway between any two of these codewords. Without any further information, we cannot determine which of these three possible codewords is closest to $[B, 0, 1]$.

Although the (3, 2, 2) code cannot correct any errors, it is capable of detecting that at least one component of $[B, 0, 1]$ was incorrect. Since the code has this capability, it is considered to be an example of an error-detecting code.

Because the (3, 2, 2) code was not sufficient to make the magic trick work, a fourth card was added. Each symbol is now encoded using the code as given in FIGURE 4. It is still true that any two components suffice to identify the symbol. The reader can verify from FIGURE 4 that each codeword differs from any other codeword in at least 3 positions. Thus, this code can be expressed using the parameters (4, 2, 3). Since an error-correcting code can correct up to $\lfloor (d-1)/2 \rfloor$ errors, this code is capable of correcting one error.

Symbol	Codeword	Symbol	Codeword
R○	UUUU = [0 , 0 , 0 , 0]	G○	DURL = [A , 0 , 1 , B]
R■	URDL = [0 , 1 , A , B]	G■	DRLU = [A , 1 , B , 0]
R◇	UDLR = [0 , A , B , 1]	G◇	DDDD = [A , A , A , A]
R★	ULRD = [0 , B , 1 , A]	G★	DLUR = [A , B , 0 , 1]
Y○	RULD = [1 , 0 , B , A]	B○	LUDR = [B , 0 , A , 1]
Y■	RRRR = [1 , 1 , 1 , 1]	B■	LRUD = [B , 1 , 0 , A]
Y◇	RDUL = [1 , A , 0 , B]	B◇	LDRU = [B , A , 1 , 0]
Y★	RLDU = [1 , B , A , 0]	B★	LLLL = [B , B , B , B]

Figure 4 A (4, 2, 3) Reed–Solomon code based on the magic trick

For example, considering the earlier example of "Up Down Up Right", $[0, A, 0, 1]$, for the four responses, now only one of the 16 possible codewords, $[0, A, B, 1]$, differs from $[0, A, 0, 1]$ in at most one position. As long as the volunteer provided at most one incorrect answer to the four cards, the responses will always differ from a valid codeword in at most one position.

In general, the problem of decoding a codeword is not easy. We need to determine both the location of the error(s) added to the codeword, as well as the correct value at each error location. In the simplified case of this magic trick, we can step through the four cards, ignoring each one in its turn, to see if the same symbol appears in the upper section of the remaining three cards. For those familiar with error-correcting codes, we are essentially "erasing" each of the four components in the codeword, one at a time. The remaining three cards form a (3, 2, 2) code. Recall that this code cannot correct any errors, but it can detect whether or not an error occurred. If we erase the correct card, then the remaining three cards should decode to a valid codeword (i.e., one of the symbols should appear in the upper section of the three remaining cards). If we erase the wrong card, then the (3, 2) code will detect the error and we should try a different erasure. Since there are only four choices for the incorrect card, this erasure decoding procedure finds the incorrect card efficiently. It should be noted, however, that this technique is not efficient for larger codes.

Reed–Solomon codes

The cards used in the trick are based on a particular finite field—that is, an algebraic system with finitely many elements, together with operations that satisfy the usual axioms for addition and multiplication. A finite field is sometimes called a *Galois field* because the research of Èvariste Galois was influential in this branch of mathematics.

Let us consider the set of four elements $\{0, 1, A, B\}$ and define the operations $\oplus$ and $\odot$ according to the arithmetic tables given in FIGURE 5. This is a finite field. For the rest of this paper, we will use the symbol $\mathbb{F}_4$ to represent the finite field with elements $\{0, 1, A, B\}$ and operations $\oplus$, $\odot$.

$\oplus$	0	1	A	B
0	0	1	A	B
1	1	0	B	A
A	A	B	0	1
B	B	A	1	0

$\odot$	0	1	A	B
0	0	0	0	0
1	0	1	A	B
A	0	A	B	1
B	0	B	1	A

Figure 5 Arithmetic tables of $\mathbb{F}_4$

We will now say what we mean by a *Reed–Solomon Code*. In practice, a code is defined by how we select its codewords, how we encode content into the codewords, and how we decode the codewords to recover the content.

Every step depends on representing vectors as polynomials. Given any vector with n entries, we associate it with a polynomial (of degree at most $n - 1$) whose coefficients are the entries of the vector. For example, if $[0, A, B]$ is a vector whose entries are elements of $\mathbb{F}_4$, we associate it with the polynomial $0x^2 + Ax + B$.

To construct a Reed–Solomon Code with parameters n and k, we first select a finite field, and then we select a particular *generating polynomial* of degree $n - k$ with coefficients from this finite field. In the case of the three-letter (3, 2, 2) code used above, we use $n = 3$, $k = 2$, the finite field $\mathbb{F}_4$, and the generating polynomial

$$g(x) = x - A.$$

For a more general Reed–Solomon Code, the generating polynomial is typically

$$g(x) = \big(x - a\big)\big(x - a^2\big)\big(x - a^3\big) \cdots \big(x - a^{n-k}\big),$$

where a is an element whose powers generate the multiplicative group of the field.

The codewords are the vectors corresponding to polynomials of degree $n - 1$ that are multiples of $g(x)$. For example, in our $(3, 2, 2)$ code, the vector $[0, A, B]$ is a codeword because its associated polynomial is a multiple of $g(x)$:

$$0x^2 + Ax + B = (x - A)(0x + A)$$

(all calculations being done in $\mathbb{F}_4$). You can see this codeword in FIGURE 3.

To encode a message, we must first represent the message as a vector of length k (the number of information components) whose entries are in the finite field. We then interpret the vector as a polynomial, the *message polynomial*, $m(x)$. Now to find the codeword for the message, we divide $m(x) \cdot x^{n-k}$ by $g(x)$ to obtain quotient $q(x)$ and remainder $r(x)$. Then the codeword $w(x)$ is given by

$$w(x) = q(x) \cdot g(x) = m(x) \cdot x^{n-k} - r(x).$$

In the magic trick, a message is one of our 16 symbols, or equivalently, an ordered pair $[c, s]$ where c is a color and s is a shape. Here, each of the four colors is represented as an element from the set $\mathbb{F}_4 = \{0, 1, A, B\}$. In this paper, 0 represents Red, 1 represents Yellow, A represents Green, and B represents Blue. Similarly, each of the four shapes is also represented as an element from $\mathbb{F}_4$. Here, 0 represents Circle, 1 represents Square, A represents Diamond, and B represents Star. For example, the message $[0, A]$ represents the Red Diamond, so that the message polynomial is

$$m(x) = 0x + A.$$

In general, the symbol $[c, s]$ becomes the polynomial $m(x) = cx + s$.

Now divide $m(x) \cdot x$ by $x - A$ using long division. In the general case of $m(x) = cx + s$, we find that

$$\begin{aligned} m(x) \cdot x &= c \cdot x^2 + s \cdot x \\ &= \big[(c \cdot x + (s \oplus (A \odot c))\big] \cdot (x - A) + \big[(c \odot B) \oplus (A \odot s)\big], \end{aligned}$$

so the quotient is $q(x) = c \cdot x + (s \oplus (A \odot c))$ and the remainder is $r(x) = (c \odot B) \oplus (A \odot s)$. Hence, the corresponding codeword associated with the message $[c, s]$ is given by

$$\begin{aligned} w(x) &= q(x)(x - A) \\ &= [\, c,\ s,\ (c \odot B) \oplus (A \odot s)\,]. \end{aligned} \tag{1}$$

Repeating this calculation for all of the possibilities for $[c, s]$, we get a $(3, 2, 2)$ Reed–Solomon code.

For example, the message polynomial associated with the Red Diamond would be $0 \cdot x + A$, since 0 represents Red and A represents Diamond. Using (1) to compute the third component, the codeword associated with the Red Diamond is $[0, A, (0 \odot B) \oplus (A \odot A)] = [0, A, 0 \oplus B] = [0, A, B]$.

To construct the first three cards, we associate each element of $\mathbb{F}_4 = \{0, 1, A, B\}$ with one of the four quadrants on each card. In this paper, the upper quadrant is associated with 0, the right quadrant is associated with 1, the bottom quadrant is associated with A, and the left quadrant is associated with B. Now place each of the 16 symbols somewhere in the correct quadrant on each of the three cards. For example, the Red Diamond, with codeword $[0, A, B]$, would be placed in the upper section of the first card, the bottom section of the second card, and the left section of the third card.

The fourth card As mentioned earlier, the $(3, 2, 2)$ Reed–Solomon code has minimum distance 2 and thus cannot correct any errors. An (n, k, d) Reed–Solomon code can be extended to an $(n + 1, k, d + 1)$ code (minimum distance $d + 1$) by appending a fourth component called a *parity check* to the codeword. In a parity check, the sum of all of the codeword components should be zero. For the magic trick, a formula that can be used to compute the parity check component for each symbol is given by $p = c \oplus s \oplus t$, where t is the third component of the codeword, i.e., $(A \odot s) \oplus (c \odot B)$. Summing the four components, we see that $c \oplus s \oplus t \oplus p = c \oplus s \oplus t \oplus (c \oplus s \oplus t) = 0$.

In the case of the Red Diamond, $t = B$, so the parity check would be computed as $p = 0 \oplus A \oplus B = 1$. The $(4, 2, 3)$ codeword associated with the Red Diamond is given by $[c, s, t, p] = [0, A, B, 1]$. The rest of the $(4, 2, 3)$ code is given in FIGURE 4. The fourth card is constructed using the parity check component for each of the codewords. Since the parity check component of the Red Diamond is 1, then the Red Diamond would be placed on the right quadrant of the fourth card.

The order of the symbols within each section of the cards was chosen so that any chosen symbol would appear in the same position within the upper section after rotation. Once the assignment of the symbols to sections was made using the $(4, 2, 3)$

Reed–Solomon code, the assignment of the symbols within the sections was implemented by logical deduction and a little trial and error.

Application to CD and DVD players

This magic trick is a significantly simplified representation of how a CD or DVD player can process data on a disc containing fingerprints or scratches. Instead of representing the data as a finite field with 4 elements, a finite field of size 256 is typically used. This finite field consists of the elements $\{0, 1, \alpha, \alpha^2, \dots, \alpha^{254}\}$ and much larger arithmetic tables.

More powerful Reed–Solomon codes are able to correct many errors per codeword. Typically, a $(255, 223)$ code with generating polynomial

$$g(x) = \big(x - \alpha\big)\big(x - \alpha^2\big)\big(x - \alpha^3\big)\cdots\big(x - \alpha^{255-223}\big)$$

is used. It can be mathematically proven that this code has minimum distance $d = 33$. Using the formula $(d - 1)/2$, we see that this code is capable of correcting up to 16 errors that may be present in the 255 components of each codeword.

Instead of the erasure decoding technique used in the magic trick, sophisticated methods such as the Berlekamp–Massey algorithm and the Euclidean algorithm are typically used to solve this problem. It is not generally known, however, that these two techniques are equivalent (see [**2**]).

REFERENCES

1. Richard Ehrenborg, Decoding the Hamming Code, *Math Horizons*, special issue on *Codes, Cryptography, and National Security*, **13** (2006) 16–17.
2. Todd D. Mateer, On the equivalence of the Berlekamp–Massey and Euclidean algorithms for algebraic decoding, *Proceedings of the 2011 Canadian Workshop on Information Theory (IEEE),* Kelowna, British Columbia, Canada, May 2011.
3. ———, A magic trick based on the Hamming Code, *Math Horizons* **21**(2) (November, 2013) 9–11, `http://dx.doi.org/10.4169/mathhorizons.21.2.9`.
4. Todd K. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*, John Wiley & Sons, New York, 2005.

Summary A magic trick based on properties of a (4, 2, 3) extended Reed–Solomon code is introduced. We then discuss relevant concepts of the Reed–Solomon code to explain why the trick works and how it was designed. This magic trick can be used as a teaching device in introductory error-correcting codes courses.

Downloadable images of the cards used in this trick appear in an article supplement at the Magazine's website. Through the end of 2014, the images can also be downloaded from `http://www.mathematicsmagazine.org`.

If you are going to perform this trick, you might want to experiment with different methods of identifying the selected symbol. With practice—and depending on your memory skills—you might be able to avoid handling the cards, or to present them in random order.

Your method of identifying the card amounts to a decoding algorithm for this Reed-Solomon code. Fast, effective decoding algorithms can be tricky, even for a very simple code!