Lecture Notes on ERROR DETECTING CODES

These notes are largely based on *Identification Numbers and Check Digit Schemes* by Kirtland.

1 Introduction

Barcodes, ISBN numbers on books, airline tickets, ebay purchase, all have a string of digits associated to them. If there is an error in them we want to certainly detect this (the point of these notes) and even correct it (a different and more complicated topic).

What are the most common types of errors?

1. A *Single-Digit-Error* occurs when one of the digits changes its values. For example if the intended string is

   9395681 and instead you have

   9395181.

   Of all transmission errors, this one occurs 79.1% of the time. Note that we are not saying that it happens that often, just that *of* the errors that happen, this is quite common.

2. A *Transposition-of-Adjacent-Digits-Error* occurs when two side-by-side digits change places. For example if the intended string is

   9395681 and instead you have

   9395861.

   Of all transmission errors, this one occurs 10.2% of the time.

3. A *Jump-Transposition-Error* occurs when two different digits, separated by a third digit between them, change places. For example if the intended string is

   93981671 and instead you have

   93961871.

   Of all transmission errors, this one occurs 0.8% of the time.

4. A *Twin Error* occurs when two identical side-by-side digits change to a different pair of identical digits For example if the intended string is

   9395571 and instead you have

   9398871.

   Of all transmission errors, this one occurs 0.5% of the time.

5. A *Phonetic Error* occurs when two digits in the number, presented orally, are recorded incorrectly. For example, a person says "Fourteen" and this is misheard as "Forty". Of all transmission errors, this one occurs 0.5% of the time. This error we are not going to deal with as it not one that Mathematics can help us with.

6. A *Jump-Twin Error* occurs when two identical digits, separated by a third digit, change to a different pair of identical digits. For example if the intended string is

   9395581 and instead you have

   8385581.

   Of all transmission errors, this one occurs 0.3% of the time.

2 Real Example of Real Error Detection Codes

This section will be examples of Error Detection Codes used in the real world, and comments on them. They all use a check digit — one digit added to the message you want to send. In all of the cases below computing the check digit will be easy. The questions of interest will be the following.

- Which of the errors mentioned above will they catch?
- If there is some error it does not catch, what is the probability that it won't catch it?
- Does the scheme work only for certain lengths of sequences, or is it extendable?

We describe the first example in more detail than the rest so that you will see how they are used.

1. US Postal Money Orders ID numbers. These are 11 digits long. The first 10 digits are the real information, but the last digit is determined as follows. Say your information is $a_1a_2\cdots a_{10}$. Then let

$$a_{11} = a_1 + a_2 + \cdots + a_{10} \pmod 9.$$

So here is the protocol the postal service uses: Bob places an order and the US postal service numbers it with number $a_1\cdots a_{10}$. They then compute a_{11} as above. Whenever they transmit the information they send $a_1a_2\cdots a_{11}$. This transmission may have an error in it. Lets say the message received is $b_1b_2\cdots b_{11}$. (The scenario we have in mind is that most of the time $a_1a_2\cdots a_{11} = b_1b_2\cdots b_{11}$, and even if this is false its only a small error, and hopefully it will be caught, as we will see.) The receiver checks if

$$b_{11} = b_1 + b_2 + \cdots + b_{10} \pmod 9.$$

If YES then the message is thought to be without error (this may be wrong, but if the scheme is good it will be wrong with a very low probability). If NO then the message is definitely wrong and she may ask that it be resent. Note that these schemes *detect* that there is an error, but do not *correct* them or even know where they are.

Def 2.1 Let X be one of the type of errors listed above.

(a) An error detection code *catches an error of type X* if whenever such an error is made then

$$b_{11} \not\equiv b_1 \times 10^{13} + b_2 \times 10^{12} + \cdots + b_{14} \times 10^0 \pmod 9.$$

(b) An error detection code *catches an error of type X with probability p* if, assuming all digits are equally likely, the probability that the error is caught is probability p. (We will see examples of this.)

Convention 2.2 We will assume the error does not occur with the check digit. In most cases the probability that this is caught is much higher then the cases analyzed.

Note 2.3 The definition of when an error is caught can be applied to any scheme with one check digit.

Theorem 2.4 *Assume we are using the Postal Money Order scheme.*

(a) The probability that a Single-Digit Error is caught is $\frac{44}{45}$.

(b) The probability that a Transposition-of-Adjacent-Digits error is caught is 0.

(c) This scheme easily extends to any number of digits.

Proof:

1) Assume that there is an i, $1 \leq i \leq 10$, such that $a_i \neq b_i$ but $(\forall j \neq i)[a_j = b_j]$. We need to show that $a_{11} \neq b_{11}$.

Consider $a_{11} - b_{11} \pmod 9$. Since $(\forall j \neq i)[a_i = b_i]$ we have

$a_{11} - b_{11} \equiv a_i - b_i \pmod 9$.

If this is 0 then the error will not be caught. This only happens if (a_i, b_i) is $(0, 9)$ or $(9, 0)$. There are 90 possible ordered pairs (a, b) with $0 \leq a, b \leq 9$ with $a \neq b$. Hence the probability that the error is not caught is $\frac{2}{90} = \frac{1}{45}$. Hence the scheme catches the error with probability $\frac{44}{45}$.

2) Assume that there is an i, $1 \leq i \leq 9$, such that $b_i b_{i+1} = a_{i+1} a_i$. Note that

$$a_{11} - b_{11} = (a_i + a_{i+1}) - (a_{i+1} + a_i) = 0.$$

Hence the error will not be detected.

3) If Postal codes were n digits long then the code can be adjusted to be

$$a_{n+1} = a_1 + a_2 + \cdots + a_n \pmod 9.$$

The same probabilities in parts 1 and 2 of this theorem would still hold. ∎

2. Airline Ticket ID Numbers. These are 15 digits long. The first 14 digits are the real information, but the last digit is determined as follows. Say your information is $a_1 a_2 \cdots a_{14}$. Then let

$$a_{15} =$$

$$a_1 + 3a_2 + 2a_3 + 6a_4 + 4a_5 + 5a_6 + a_7 + 3a_8 + 2a_9 + 6a_{10} + 4a_{11} + 5a_{12} + a_{13} + 3a_{14} \pmod 7.$$

Theorem 2.5 *Assume we are using the Airline Ticket ID Number scheme.*

(a) The probability that a Single-Digit-Error is caught is $\frac{14}{15}$.

(b) The probability that a Transposition-of-Adjacent-Digits error will be caught is $\frac{14}{15}$.

(c) This scheme easily extends to any number of digits.

Proof:

1) Assume that there is an i, $1 \leq i \leq 14$, such that $a_i \neq b_i$ but $(\forall j \neq i)[a_j = b_j]$.

Consider $a_{15} - b_{15} \pmod 7$. Since $(\forall j \neq i)[a_i = b_i]$ we have

$a_{15} - b_{15} \equiv 10^i (a_i - b_i) \pmod 7$.

This will be 0 (and hence the error undetected) if $a_i - b_i \equiv 0 \pmod 7$. This happens when (a_i, b_i) is any of

$(0, 7), (1, 8), (2, 9), (7, 0), (8, 1), (9, 2)$.

This is 6 ordered pairs out of a possible 90. Hence the probability of not being caught is $\frac{6}{90} = \frac{1}{15}$. Thus the probability of being caught is $\frac{14}{15}$.

2) Assume that there is a Transposition-of-Adjacent-Digits error on the ith place. This means that $a_i a_{i-1} = b_{i-1} b_i$.

Lets look at $a_{15} - b_{15} \pmod 7$. Most of the terms cancel out (since all of the other a_j's equal the b_j's).

$$a_{15} - b_{15} \equiv (a_i 10^i + a_{i-1}10^{i-1}) - (a_{i-1}10^i + a_i 10^{i-1})$$

$$\equiv 10^{i-1}(10a_i + a_{i-1} - 10a_{i-1} - a_i) \equiv 10^{i-1}(9a_i - 9a_{i-1}) \equiv 2 \times 10^{i-1}(a_i - a_{i-1}).$$

Since 7 is prime this only happens when $a_i - a_{i-1} \equiv 0 \pmod 7$. As in part 1, the probability of the error being caught is $\frac{14}{15}$.

3. If Airlines used n digit codes then the scheme can be generalized. We leavet his to the reader. ∎

4. Bill's Collection of *Funny t-shirts* are assigned ID numbers. These are 12 digits long. The first 11 digits are the real information, but the last digit is determined as follows. Say your information is $a_1a_2 \cdots a_{11}$. Then let

   $a_{12} = a_1 + a_2 + \cdots + a_{11} \pmod{10}$.

   Theorem 2.6 *Assume we are using the T-shirt Scheme.*

   (a) All Single-Digit-Errors are caught.

   (b) The probability that a Transposition-Of-Adjacent-Digit-Error will be caught is 0.

   (c) This scheme easily extends to any number of digits.

   Proof:

   1) Assume that there is an i, $1 \leq i \leq 11$ such that $a_i \neq b_i$ but $(\forall j \neq i)[a_j = b_j]$.

   Consider $a_{12} - b_{12} \pmod{10}$. Since $(\forall j \neq i)[a_i = b_i]$ we have

   $a_{12} - b_{12} \equiv (a_i - b_i) \pmod{10}$.

   This will never be 0. Hence the error is always detected.

   2) Assume that there is an i, $1 \leq i \leq 9$, such that $b_ib_{i+1} = a_{i+1}a_i$. Note that

   $$a_{11} - b_{11} = (a_i + a_{i+1}) - (a_{i+1} - a_i) = 0.$$

   Hence the error will not be detected.

   3) If Bill acquires more T-shirts and needs to extend to n digits then he can use the following scheme.

   $a_{n+1} = a_1 + a_2 + \cdots + a_n \pmod{10}$.

   ∎

5. Universal Product Codes (barcodes). These are 12 digits long. The first 11 digits are the real information, but the last digit is determined as follows. Say your information is $a_1a_2 \cdots a_{11}$. Then let

   $a_{12} = (a_1 + 3a_2 + a_3 + 3a_4 + \cdots + a_9 + 3a_{10} + a_{11}) \pmod{10}$.

   Theorem 2.7 *Assume we are using the Universal Product Code Scheme.*

   (a) All Single-Digit-Errors are caught.

   (b) The probability that a Transposition-of-Adjacent-Digits-Errors will be caught is 8/9.

   (c) This scheme easily extends to any number of digits.

Proof:

1) Since this uses base 10 this is similar to the T-shirt codes.

2) Assume there is a Transposition-Of-Adjacent-Digits-Error. Hence there is an i, $1 \leq i \leq 10$, with $a_i a_{i+1} = b_{i+1} b_i$. We will assume that i is odd (the case of i even is similar).

$$a_{12} - b_{12} \equiv (a_i + 3a_{i+1}) - (3a_i + a_{i+1}) = 2(a_{i+1} - a_i) \pmod{10}.$$

This is zero when $a_{i+1} - a_i = 5$. This happens when

$$(a_i, a_{i+1}) \in \{(0,5), (5,0), (1,6), (6,1), (2,7), (7,2), (3,8), (8,3), (4,9), (9,4)\}.$$

So what happens for 10 ordered pairs out of the 90, which is 1/9. Hence the probability that the scheme catches this error is 8/9.

3) If we need n digit code then we can use the following.

If n is odd then we can extend via

$$a_{n+1} = (a_1 + 3a_2 + a_3 + 3a_4 \cdots + a_{n-2} + 3a_{n-1} + a_n) \pmod{10}.$$

If n is even then we can extend via

$$a_{n+1} = (a_1 + 3a_2 + a_3 + 3a_4 \cdots + 3a_{n-2} + a_{n-1} + 3a_n) \pmod{10}.$$

∎

6. ISBN numbers for books. This scheme uses mod 11. Hence a symbol is needed for the 'digit' 10. They use X. These are 10 digits long. The first 9 digits are the real information, but the last digit is determined as follows. Say your information is $a_1 a_2 \cdots a_{10}$. Then let

   $a_{10} = -(a_1 + 2a_2 + 3a_3 + 4a_4 \cdots + 9a_9) \pmod{11}$.

   Note that a_{10} could be 10 which we denote by X.

Theorem 2.8 *Assume we are using the ISBN Scheme.*

(a) All Single-Digit-Errors are caught.

(b) All Transposition-of-Adjacent-Digits-Errors are caught.

Proof:

1) Assume that there is an i, $1 \leq i \leq 9$ such that $a_i \neq b_i$ but $(\forall j \neq i)[a_j = b_j]$.

Consider $a_{10} - b_{10} \pmod{1}1$. Since $(\forall j \neq i)[a_i = b_i]$ we have

$a_{10} - b_{10} \equiv (ia_i - ib_i) \equiv i(a_i - b_i) \pmod{1}1$.

Since 11 is prime and larger than 10, and $i \neq 0$ can only be 0 if $a_i = b_i$. Hence this is never 0. Therefore the error will be detected.

2) Assume there is a Transposition-Of-Adjacent-Digits-Error. Hence there is an i, $1 \leq i \leq 8$, with $a_i a_{i+1} = b_{i+1} b_i$.

$$a_{10} - b_{10} \equiv (ia_i + (i+1)a_{i+1}) - ((i+1)a_i + ia_{i+1}) = (a_{i+1} - a_i) \pmod{11}.$$

Since $11 > 10$ this only happens when $a_{i+1} = a_i$. Hence this is never 0. Therefore the error will be detected. ∎

Note that ISBN numbers use a funny new symbol X and cannot be extended to n digits.

Thought question: is there a scheme that

1. Uses one Check Digit.
2. Does not use any additional symbols.
3. Is extendable.
4. Catches all Single-Digit-Errors.
5. Catches all Transposition-of-Adjacent-Digits Errors.

3 Hashing Codes

For some applications you MUST have unique ID's. Here is a real example.

For drivers licenses the Government wants to take information about you and make it into a number. The scheme to do this should be simple, but also not lead to two people mapping to the same number. If the scheme is only based on (say) the last name then EVERYONE with the same last name maps to the same number. If the scheme is based on first name and last name this is better unless your name is John Smith. We will look at ways to map names to numbers that try to minimize collisions.

We look at how Washington State actually assigns drives license numbers. We will use Joseph H. Kirtland, who was born on Jan. 1, 1978, for our example. The code is 12 symbols long — some are numbers, some are letters.

1. The first five characters are the first five letters of of the last name. In our case KIRTL. If the name was less than five letters, say the last name was Fee, then the rest is filled out with asterisks. So in that case it would be FEE**.
2. The sixth character is the individuals first initial. In our case J. So we have KIRTLJ. If the person has no first name, use a *.
3. The seventh character is the individuals middle initial. In our case H. So we have KIRTLJH. If the person has no middle initial, use a *. (Note- some systems may use an N as some insert NMI (no middle initial) if there is no middle initial.)
4. The eight and ninth character is obtained by subtracting the last two digits of the year of birth from 100. In our case $100 - 78 = 22$. So we have KIRTLJH22.
5. The 10th character is the check digit. This is determined as mentioned in the last section. For now we will denote in a_{10}.
6. The 11th character is the month-of-birth code. Here is how we map Months to symbols and also an alternative that will be used later to avoid collisions.

   Since Joseph H. Kirtland was born on in Jan his code is B. So we have KIRTLJH22a_{10}B so far.
7. The 12th (and last) character is the code for the day of the month the subject is born. Here is how the code works.
   (a) t1-A, 2-B, 3-C, 4-D, 5-E, 6-F, 8-H.
   (b) (We do not use I since I and 1 are easily confused.) 9-Z, 10-S, 11-J, 12-K, 13-L, 14-M, 15-N, 16-W, (We do not use O since O and 0 are easily confused.) 17-P, 18-Q, 19-R.
   (c) 20-0 (this is zero), 21-1, 22-2, ..., 29-9 (This is nice since these get mapped to what they are mod 10.)
   (d) 30-T, 31-U.

Month	Code	Alternative Code
Jan	B	S
Feb	C	T
Mar	D	U
Apr	J	1
May	K	2
Jun	L	3
Jul	M	4
Aug	N	5
Sep	O	6
Oct	P	7
Nov	Q	8
Dec	R	9

Since Kirtland was born on the 1st, his code is A. So his number is KIRTLJH22a_{10}BA. We leave it as an exercise to find a_{10}. Assume this has been done.

8. We are not done. We check if someone else with the exact same license number is already in the database. (Perhaps Joseph H. Kirtland's cousin Josepphine H. Kirtland, who by accident is born on the same day. Or perhaps Joseph's twin brother.) If someone else has the same license number then we use the ALTERNATIVE for the Month-born code.