Error Detection in Codes

Hugh Chipman & Catherine Stanley,

Acadia University Department of Mathematics & Statistics
hugh.chipman@acadiau.ca

This document contains presentation material. Instructor notes are at the end of the document.

Error detection in codes

0 66741 11003 3

ACADIA
UNIVERSITY

Mathematics and Statistics
Hugh Chipman & Catherine Stanley

Have you ever played the game “telephone”?


[1]

One person whispers a message in the next person’s ear. Keep passing it on. Often the final message is quite different.

Can we tell if a message has been mixed up?

Similar problems happen with numeric codes that contain information:

- Skype calls
- Music or video
- Data sent over the internet
- Barcodes on products
- QR codes

A game[2]:

- Lay out a 5 x 5 grid of black and white tiles in any pattern you want.
- I will look at the tiles
- When I’m not looking, you will flip one tile.
- I have to guess which tile it is.

[2] From https://classic.csunplugged.org/error-detection/ , shared under a Creative Commons BY-NC-SA 4.0 licence.

(This page intentionally blank)

How did I guess correctly?

When I added an extra row and column, I chose the color of each square carefully, so that every row and every column had an even number of black tiles.

What happens when you then flip one tile?

This is an example of an “error correcting code”. The data has a built-in check which can help identify errors.

You try it!

1. One person lay out the cards 5 × 5.
2. How many grey cards are there in each row and column? Is it an odd or even number?
3. Now add a sixth card to each row, making sure the number of grey cards is always even.
4. Add a sixth row of cards along the bottom, to make the number of grey cards in each column an even number.
5. Now flip a card. What do you notice about the row and column?
6. Now take turns to perform the ‘trick’.

What kind of errors can this detect?

What kind of errors can this NOT detect?

Checksum for binary numbers

A similar check can work for a sequence of binary (0/1) numbers (instead of a square with 2 colors).

- Suppose I want to send a “10011”.
- I add a 6th digit on the right to give an even number of “1”s: 10011<u>1</u>

So you can tell there’s a mistake here: 11011<u>1</u>

Can we tell which digit is wrong if we just see "110111"?

The last “1” is called a “checksum”.

Another kind of error detecting code: barcodes

A “UPC” barcode is a “universal product code”

The last digit is a checksum.

066741110033

But the rule for making the checksum is different.

Another kind of error detecting code: barcodes

Step 1: Add the digits in the odd numbered positions, and multiply the total by 3:

0+6+4+1+0+3 = 14

14 x 3 = 42

Step 2: Add the digits in the even numbered positions to the result:

(6+7+1+1+0)+42 = 57

Step 3: The checksum is the digit that, when added to the total, gives a result divisible by 10:

57 + _ = _

So the code

066741110033

is correct because…

```
(0+6+4+1+0+3) x 3 = 14 x 3 = 42
 066741110033
  6+7+1+1+0+3      = 18       +18
                              =60
```

This is a convenient way to arrange the calculations. Notice that the checksum is in an even location, so we can just add it at the end also.

To make our calculations easier, we will work with 6-digit numbers instead of 12.

For example, the code

066747

is correct because…

```
(0+6+4) x 3 = 10 x 3 = 30
 066747
  6+7+7     = 20      +20
                      ___
                      =50
```

You try it!

Pick a digit, and try replacing it with a wrong digit. Verify that when you change the digit, the result of the calculation

(3*odd + even)

is no longer divisible by 10.

Can we figure out why this works?

Why it works

Consider a digit in an even location. What happens to the result if this digit gets bigger by 1? Smaller by 1?

```
(0+6+4)  x  3  =  10  x  3  =  30
 066747
   6+7+7         =  20           +20
                                 =50


(0+6+4)  x  3  =  10  x  3  =  30
 066 47
   6+ +7         =               +..
                                 =50
```

Is there any digit I could change the "7" to, and still get a result of 50?

NOTE: Instead of a " ", we could use a letter ("z") to represent the new digit.

If you've learned algebra, you've seen this sort of thing.

With algebra, you can prove 7 is the only answer:

```
(0+6+4)  x  3  =  10  x  3  =  30
  066z47
    6+z+7           =                   +..
                                        =50
```

(fill in the blanks above. Can you solve for "z"?)

Why it works (continued)

OK, that wasn't so bad.

Can we change a digit in an odd location?

```
(0+6+4) x 3 = 10 x 3 = 30
 066747
   6+7+7    = 20        +20
                        ___
                        =50


(0+6+ ) x 3 =    x 3 =
 0667 7
   6+7+7    = 20        +20
                        ___
                        =__
```

(In this case, we could get other numbers ending “0” besides 50)

Same as above, but with Algebra:

(0+6+z) x 3 =

0667z7

6+7+7 = 20

+20

= __

More errors?

We just showed that the UPC barcode can always detect an error in one digit.

Can it detect a "switch" of two adjacent digits?

Try switching 6 & 7 below…..

(0+6+4) x 3 = 10 x 3 = 30

0 6 67 4 7

6+7+7 = 20 +20

=60

(0+7+4) x 3 =

0 6 76 4 7

6+6+7 =

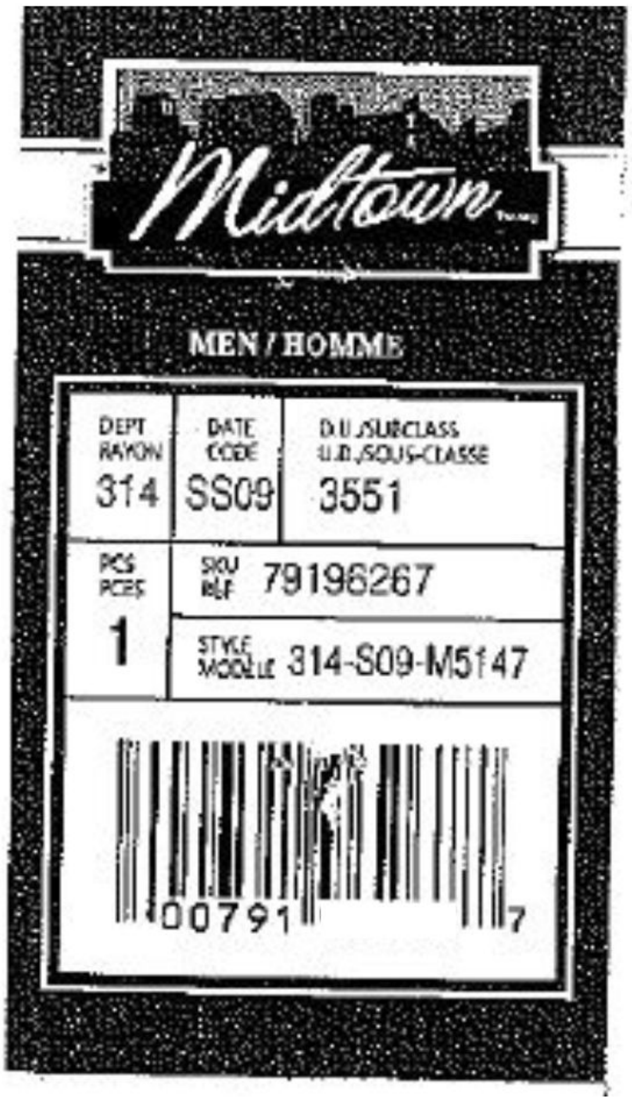
More errors?

So it looks like the checksum can detect swapped adjacent digits (a common error for humans copying numbers). In fact there are a few possible pairs that would go undetected.

Challenge: Can you find two such numbers?

Easier question: Can you switch non-adjacent digits and leave the answer unchanged?

Which code (or codes) are correct in the 3 pictures below?

<table>
<tr>
<td>
</td>
<td>67383 0</td>
<td>
</td>
</tr>
<tr>
<td>007917</td>
<td>673830</td>
<td>189996</td>
</tr>
</table>

So, when the cashier scans an item and it doesn't register, and he has to scan it again, that may be because the checksum digit didn't add up correctly.

Bonus question:

Our rule here was

1. add digits at odd locations and multiply by 3
2. add digits at even locations
3. add answers from part 1 and 2, and add the checksum
4. The result must be divisible by 10.

What if we multiplied by "2" instead of "3" in the first step? Would that work as well?

What happens when you multiply the digits 0 to 9 by a number from 0 to 9 (we’re multiplying by 1 and 3 here)?

0 1 2 3 4 5 6 7 8 9

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ multiply by 1

0 1 2 3 4 5 6 7 8 9

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ multiply by 2

0 1 2 3 4 5 6 7 8 9

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ multiply by 3

Since only the last digit matters when we ask if something is divisible by 10, is multiplying by 2 as good a choice as multiplying by 3?

It turns out there are 2 other digits besides 1 and 3 that you could multiply by, and get just as good results. What are they?

By the way, real barcodes are 12 digits, not 6. But the idea is the same.

There is also a checksum digit in ISBN-10 numbers, inside all your books. For example the ISNB-10 for “The End” by Lemony Snicket is

0-064-41016-1

The checksum digit (the last one) is chosen so that the following calculation gives a multiple of 11:

10x**0**+9x**0**+8x**6**+7x**4**+6x**4**+5x**1**+4x**0**+3x**1**+2x**6**+1x**1**
= 121 = 11x11

The End (really)

INSTRUCTOR NOTES

This unit explores the math behind "error correcting codes".

- Binary checksum material assumes students know about binary number systems. Only enough background is needed to know that you can store any number as a sequence of 0s and 1s.
- In the “game” at the start,
  - the teacher has 36 tiles, which are (for example) black on one side, and white on the other.
  - For later in this activity, you should have paper versions of the same game, each with 36 tiles. These could be roughly 2cm square, with white/black or X/O put on the tiles, and cut out. For every pair of students you need one set of 36 tiles.
  - Get students to lay out a 5 by 5 pattern of black/white tiles.
  - Teacher adds a 6th row and column "just to make it more challenging".
  - The extra tiles are added so that every row and column has an even number of white tiles.
  - Teacher then turns away and tell students to flip one card.
  - It will be possible to identify the card, by looking for the row and column with an odd number of tiles
- After a few repetitions, explain how you did it, and break students into groups of 2, giving each group their own set of tiles
- The main example of a checksum (barcodes), uses modular arithmetic (just a bit).

- The checksum rule is more complicated, but still easy enough for the kids to try out.
- The method of copying numbers in odd/even positions above/below the original string helps them do the calculations.
- 6-digit numbers used instead of the "real" 12-digit codes, to reduce tedious calculations.
- Students who have had algebra can use that approach to prove that you can't change a single digit without detection.
- Material at the end added to illustrate why 2 is not a good multiplier
- A Googe Doc version of this presentation is available at https://docs.google.com/document/d/1e2HrppjSurc4XPWKCQYKbS4qKYIIsQvxquyQQYW1zZs/edit?usp=sharing