An Alternative to Runlength-limiting Codes: Turn Timing Errors into Substitution Errors

David J. C. MacKay
Department of Physics, University of Cambridge
Cavendish Laboratory, Madingley Road,
Cambridge, CB3 0HE, United Kingdom.
mackay@mrao.cam.ac.uk

September 2 2000 — DRAFT 1.2

Standard runlength-limiting codes – nonlinear codes defined by trellises – have the disadvantage that they disconnect the outer error-correcting code from the bit-by-bit likelihoods that come out of the channel.

The normal motivation for runlength limits is to prevent loss of synchronization between the transmitter and receiver. I suggest mapping the user data to transmitted symbols in such a way that timing errors lead not to errors of synchronization, but to symbol errors.

In magnetic recording channels with high bit densities, there may be constraints not only on the maximum runlength but also on the number of successive transitions permitted. The coding method proposed here is compatible with these constraints.

1 Magnetic recording constraints

I will assume the following properties of high bit-density magnetic recording.

1. Transitions from one magnetization to another, as in '00001111', produce a strong and reliable signal. The timing of the transition event may not be reliably conveyed, so there is potential for 00001111 to be received as 00000111 or 00011111. These shifts of transition timing can be viewed as substitution errors, and do not involve loss of synchronization.

2. If the interval between two changes in magnetization is large then synchronization errors may occur. For example, 000111111000 might be interpreted as 0001111111000. Synchronization errors normally are catastrophic events, since traditional error-correcting codes are designed to detect and correct only substitutions. This possibility motivates the use of runlength-limiting codes.

3. If many transitions occur close together (for example, 0101010), then substitution errors are more likely to occur. Pairs of transitions can be lost, so that 0101 is received as 0111 or 0011. For this reason, additional constraints may be included, forbidding the transmission of sequences such as 010 and 101.

4. Insertions lengthen runs between transitions; they are not expected to *increase* the number of transitions.

The properties of this channel are reminiscent of neuronal spikes (action potentials): neuronal spikes, like magnetic transitions, are strong signals; the exact time at which a spike occurs may be affected to random influences; and it is not possible for two spikes to occur in rapid succession.

It has been proposed that the brain might convey information in the relative timing of the spikes (MacKay and McCulloch, 1952; Hopfield, 1995). In this note, I suggest the application of this idea to magnetic recording channels.

2 A simple code that needs no runlength constraints

Consider a high-density magnetic recording channel. Imagine that the minimum permitted spacing between transitions is 2, *i.e.*, the sequences 010 and

101 are forbidden. And imagine that there may be synchronization errors in runs of any length.

However, if we map source symbols to transmitted sequences by *mapping each source symbol to a different inter-transition interval*, then timing errors will be turned into *symbol* errors. The following table shows the mapping from 4 source symbols to binary transmissions assuming the previous bit sent was a 0.

Source	Transmitted	Runlength representation
a	11	2
b	111	3
c	1111	4
d	11111	5

For example, the string abb is encoded as 11000111.

This code has the following properties:

1. It is very easy to encode and decode.

2. Timing errors will cause symbol errors, especially between c and d, and between b and c.

3. The code is a variable length code. The mean encoded length of a 4000 bit source file is 7,000 transmitted bits. The rate is $2/3.5 = 4/7 = 0.57$. The worst case maximum encoded length is 10,000 transmitted bits. Assuming random source data, the encoded length is approximately Gaussian-distributed, 7000 ± 82. The probability that the encoded length would exceed 7400 is very small indeed.

2.1 Two fixes for the slight variability in blocklength

One could implement this code as a rate $4/7.4 = 0.54$ code by adding each user block to a random, sector-dependent coset vector; in the unlikely event that the resulting encoded blocklength exceeds 7400 bits, we can skip to the next sector, where the coset vector is different.

Alternatively, a very easy fix for the above code is to lengthen the transmission by one bit, and, in the event that the encoded blocklength is bigger than the average, 7000, replace all occurrences of a by d, b by c, and vice versa. The resulting blocklength will now be smaller than average. Which of the two codes is used is indicated by the single extra bit. The code would be almost exactly a rate 4/7 code. The cost of this fix is a drop in rate of 4000/4001.

3 Fancier versions of the same idea

The optimal way to use the channel with a constraint on the spacing between successive channels is for the different runlengths to be used with an exponential distribution.

$$P(l) = 2^{-Cl}, \tag{1}$$

where C is the capacity of the channel, which satisfies the implicit equation

$$Z = \sum_{\text{permitted values of } l} 2^{-Cl} = 1. \tag{2}$$

The above simple code crudely approximates this exponential by a uniform distribution over $l = 2, 3, 4, 5$.

In the case where the permitted values are $l = 2, 3, 4, 5$, these two distributions are:

l	Ideal distribution	Crude distribution
2	0.426	0.25
3	0.278	0.25
4	0.182	0.25
5	0.119	0.25

and the capacity of the channel is about 0.615.

We can improve on the simple code in two ways.

1. Make the code more complicated so as to better match the ideal exponential distribution.

2. If the variability of the encoded length is an issue, modify the encoding rule so that the variability is reduced.

3.1 Longer symbols

Another code, with 16 source symbols.

Source	Transmitted	Total length
a	5	5
b	2,3	5
c	2,4	6
d	2,5	7
e	3,2	5
f	3,3	6
g	3,4	7
h	3,5	8
i	4,2	6
j	4,3	7
k	4,4	8
l	4,5	9
m	2,2,2	6
n	2,2,3	7
o	2,2,4	8
p	2,2,5	9

The mean symbol duration is $109/16 = 6.8125$ transmitted bits. The rate of the code is $64/109 = 0.5872$.

The variability of length of this code is smaller than that of the first code. If a guarantee on the maximum blocklength is required, the 'single bit overhead' trick can be used, pairing each source symbol that has a shorter-than-average encoded length with one that has a longer-than-average encoded length. The effect of such a pairing is indicated below.

Reordered code: to make the alternative code, turn the list upside down.

Source	Length	Length in reversed code	Average length
a	5	9	14/2
b	5	9	14/2
e	5	8	13/2
c	6	8	14/2
f	6	8	14/2
i	6	7	13/2
m	6	7	13/2
g	7	7	14/2
d	7	7	14/2
j	7	6	13/2
n	7	6	13/2
k	8	6	14/2
h	8	6	14/2
o	8	5	13/2
p	9	5	14/2
l	9	5	14/2

What is a compact statement of this trick's effect? In the worst case, a document full of gs, the encoded length would be 7 per source symbol. So a file of 4000 bits gets encoded into 7000 bits, guaranteed. In terms of worst-case performance, therefore, this scheme has no advantage over the simpler code. But it does have a smaller expected length.

3.2 Longer runs

In practice, we are probably interested in codes allowing longer runs. Here is a third code, with 16 source symbols.

Source	Transmitted	Total length
a	2,2	4
b	2,3	5
c	2,4	6
d	2,5	7
e	2,6	8
f	2,7	9
g	3,2	5
h	3,3	6
i	3,4	7
j	3,5	8
k	3,6	9
l	3,7	10
m	4	4
n	5	5
o	6	6
p	7	7

This code has mean length $106/16 = 6.625$, slightly (3%) shorter than the second code.

Acknowledgements

This work was supported by the Gatsby Foundation and by a partnership award from IBM Zürich research laboratory.

References

Hopfield, J. J. (1995) Pattern-recognition computation using action-potential timing for stimulus representation. *Nature* **376** (6535): 33–36.

MacKay, D. M., and McCulloch, W. S. (1952) The limiting information capacity of a neuronal link. *Bull. Math. Biophys.* **14**: 127–135.