

Robust error correction in infofuses

Greg Morrison, Sam W. Thomas III, Christopher N. LaFratta, Jian Guo, Manuel A. Palacios, Sameer Sonkusale, David R. Walt, George M. Whitesides and L. Mahadevan

Proc. R. Soc. A 2012 **468**, doi: 10.1098/rspa.2011.0316 first published online 28 September 2011

Supplementary data

["Data Supplement"](#)

<http://rspa.royalsocietypublishing.org/content/suppl/2011/09/28/rspa.2011.0316.DC1.html>

References

[This article cites 13 articles, 1 of which can be accessed free](#)

<http://rspa.royalsocietypublishing.org/content/468/2138/361.full.html#ref-list-1>

Subject collections

Articles on similar topics can be found in the following collections

[algorithmic information theory](#) (1 articles)

[materials science](#) (118 articles)

Email alerting service

Receive free email alerts when new articles cite this article - sign up in the box at the top right-hand corner of the article or click [here](#)

Robust error correction in infofuses

BY GREG MORRISON^{1,*}, SAM W. THOMAS III^{2,4}, CHRISTOPHER N. LAFRATTA^{4,6}, JIAN GUO⁵, MANUEL A. PALACIOS⁴, SAMEER SONKUSALE⁵, DAVID R. WALT⁴, GEORGE M. WHITESIDES² AND L. MAHADEVAN^{1,3,*}

¹*School of Engineering and Applied Sciences,* ²*Department of Chemistry and Chemical Biology,* and ³*Department of Physics, Harvard University, Cambridge, MA 02138, USA*

⁴*Department of Chemistry,* and ⁵*Department of Electrical Engineering, Tufts University, Medford, MA 02155, USA*

⁶*Department of Chemistry, Bard University, Annandale-on-Hudson, NY 12504, USA*

An infuse is a combustible fuse in which information is encoded through the patterning of metallic salts, with transmission in the optical range simply associated with burning. The constraints, advantages and unique error statistics of physical chemistry require us to rethink coding and decoding schemes for these systems. We take advantage of the non-binary nature of our signal with a single bit representing one of $N=7$ states to produce a code that, using a single or pair of intensity thresholds, allows the recovery of the intended signal with an arbitrarily high recovery probability, given reasonable assumptions about the distribution of errors in the system. An analysis of our experiments with infofuses shows that the code presented is consistent with these schemes, and encouraging for the field of chemical communication and infochemistry given the vast permutations and combinations of allowable non-binary signals.

Keywords: infofuse; error-correcting codes; chemical communication

1. Introduction

Infochemistry is an emerging field, attempting to develop methods of storage and transmission of information using chemical or material means. The advantage of these modes of communication over electronic communication will depend on the speed, reliability and versatility of the transmission, as well as the conditions under which the signal is to be sent (e.g. with no power source available), but remains relatively unexplored, as the maximum rate at which information can be transmitted is limited by the physical length scales and time scales in the system, as well as the noisiness of the channels, which clearly differs significantly from their electronic analogues.

*Authors for correspondence (gmmorriso@seas.harvard.edu; lm@seas.harvard.edu).

Electronic supplementary material is available at <http://dx.doi.org/10.1098/rspa.2011.0316> or via <http://rspa.royalsocietypublishing.org>.

Recent work has shown that it is possible to transmit and receive messages by both chemical and material means with an infofuse (Thomas *et al.* 2009; Kim *et al.* 2010)—a combustible system in which patterned metallic salts encode information, and an infobubble (Hashimoto *et al.* 2009)—a bubble-based microfluidic device in which optical pulses encode information. Infofuses are advantageous in a number of ways: sending a message does not require a separate power supply, and the lightweight, compact and self-contained nature of the fuse allow for great mobility for the sender of the signal (although electrical power is still required to receive). Individual pulses from the metallic salts has extremely narrow spectral widths (0.01 nm) and enables the use of bandpass filters with the narrowest possible spectral widths (1 nm). These narrow filters allow very weak signals to be discerned over the background (Scoog *et al.* 2007). However, reliable transmission using any communication system requires the development of methods to overcome the noise in transmission, which is itself a function of the system. Over the past half century, various error-correcting codes (Peterson & Weldon 1972; Cover & Thomas 1991; Wicker & Bhargava 1994; Cohen *et al.* 1997; MacKay 2003; Mitzenmacher 2009) have been developed and allow for the possibility of accurate signal recovery with minimal loss of information density, for both binary and non-binary systems. In its most elemental form, error correction is accomplished by including redundant check bits that convolve the positions and values of each bit in the signal in a simple manner. While binary communication schemes are the most commonly studied (Hamming 1950; Peterson & Weldon 1972), non-binary alphabets (where each bit can attain one of N possible values) may increase the efficiency of these redundant bits in correcting errors (Mitzenmacher 2006, 2009) at the cost of introducing further complexity in both encoding or correction. Here, we focus our attention on the infofuse that uses a triplet code of pulses with a bit taking on $N = 7$ possible values and develop a simple error-correcting code tailored to correct the errors that take advantage of the inherent non-binary nature of the system (Mitzenmacher 2006, 2009; rather than the $N = 2$ binary signals in common digital communication approaches). The coding scheme introduces redundancy in the transmitted signal, with a message of length $n + m$ transmitted to communicate an intended signal of length n using m redundant check bits. Our approach, which combines theory and experiment, shows that this code can recover the intended sequence with near certainty, given some simple but reasonable assumptions about the distribution of insertion errors. We further show that using a pair of thresholds, dividing the data into ‘clear’ and ‘indeterminate’ peaks can increase both the reliability of recovery and reduce the computational complexity of the algorithm. Finally, we show that for the infofuse with a bit taking on $N = 7$ possible values, we achieve a good balance between the length of the fuse required to send a message and the efficiency (the ratio $n/(n + m)$) of the error correction.

2. The infofuse

(a) *The experimental system*

Infofuses (Thomas *et al.* 2009) are strips of a flammable polymer (nitrocellulose) patterned with spots of thermally emissive salts. An ignited infofuse supports a flame front that propagates along the strip at a roughly constant velocity

and successively ignites each spot in turn, thereby emitting optical pulses in time. Infofuses use three distinct alkali metals with very sharp emission spectra: potassium (K, at 766.49 nm), rubidium (Rb, at 780.03 nm) and caesium (Cs, at 852.11 nm; Ralchenko *et al.* 2010). The nitrocellulose strips are of the order of 2 to 3 mm wide, and 0.1 mm thick. The speed of the flame front as it propagates along the fuse depends strongly on the fuse width, but is generally in the range of 1–3 cm s⁻¹. The emission from each of the chemical spots is observed by a telescopic receiver that monitors the emission midpoint of each element. For a sharply defined flame front propagating along a thin fuse, the separation between the applied spots must be larger than the width of the flame, else two spots will ignite simultaneously (which will prevent the transmission of the message). Variations in the shape of the flame front, the finite width of the fuse and non-uniformities in the nitrocellulose will require a yet larger separation between peaks to enable reliable communication. In our experiments, the observed light pulses have a duration of about 100 ms (approx. 0.3 mm wide), and the spacings between subsequent chemical spots are of the order of approximately 1 cm. This separation is significantly larger than the 2–3 mm flame front, and we do not find significant overlap between the emissions from individual spots experimentally.

The detector has excellent range, and clear signals are obtainable from more than 500 m away (with an estimated 1.4 km maximum range). Signals sent within the laboratory (close range of approx. 20 m only) allowed four to five pulses per fuse before falling out of the range of view of the telescope, so long messages sent within the laboratory were broken into multiple fuses. In order to simulate the effect of outdoor conditions for experiments in the laboratory, an optical density (OD) 2 filter was inserted into the telescopic detection device. This filter reduced the incoming light intensity by approximately 99 per cent and simulated the conditions of the infofuse being about 10 times further away. Experiments performed outdoors showed that while detection of the signal was difficult in direct sunlight, if the infofuse was kept out of the sun, the signal was sufficiently above background to be detected during the day. For the purpose of encoding information with these three emitters, we used a scheme that assigns alphanumeric characters to simultaneous combinations of unique optical pulses (Thomas *et al.* 2009): seven ($2^3 - 1$) unique optical pulses exist for three distinct emitters. Each pulse combination is given a numerical value, with K = 0, Cs = 1, Rb = 2, K–Cs = 3, K–Rb = 4, Cs–Rb = 5 and K–Cs–Rb = 6. Two consecutive pulses (giving a total of 49 unique pair combinations) are therefore sufficient to encode each alphanumeric character and some special characters (see the electronic supplementary material for further discussion). This encoding scheme was used in experiments in the laboratory (with the OD 2 filter), while outdoor experiments were intended for benchmarking and did not use an encoded message.

A portable infofuse detection system, in the form of a three-channel hard-limiter receiver, was implemented for long-distance experiments to verify the single-threshold encoding/correction algorithm (figure 1). The main modules of the system consist of three channels of high-sensitivity photo-receivers (PDF10A, Thorlabs Inc.) and peripheral optics that separate and amplify three spectrum emissions from the burning infofuse (767, 78 and 852 nm). A custom printed circuit board is fabricated to perform signal conditioning prior to the error-correction algorithm. The analogue waveforms from optical receivers are first scaled to 5 V by a resistive divider, then a tunable eighth-order Butterworth

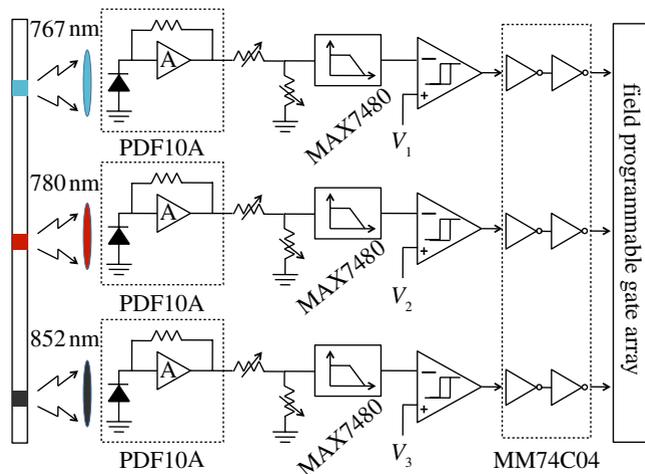


Figure 1. Block diagram of portable detection system. (Online version in colour.)

low-pass filter (LPF; MAX7480 from Maxim IC Inc.) with a cut-off frequency programmed to be around five times the data rate of the infuse is used to reduce glitches and spikes in the waveforms. After the LPF, a multi-channel voltage comparator (TLC3704 from Texas Instruments Inc.) generates event trigger digital pulses, indicating the existence of certain spectrum emissions. This is performed by comparing the output from each LPF with a predefined reference voltage (V_1 – V_3). Each reference is independently adjustable to compensate for different sensitivities and incident intensities in each optical channel. Because of the LPF, the spurious fluctuations in the digital output can be reduced. The digital outputs of the comparators are then transmitted via digital buffers (MM74C04, Fairchild Semiconductor Corp.) to a high-speed field-programmable gate array chip for digital signal processing, where an asynchronous algorithm implements message acquisition and decoding. The algorithm puts the digital system into standby mode and only wakes it up for data acquisition if the optical–electrical signal of any of the three photo-receivers crosses the thresholds V_1 – V_3 . In this way, the data-acquisition rate can be automatically adjusted to fit the speed of burning in the infuse, thus improving the data-transmission efficiency and reducing system power consumption. The results of the decoded messages are displayed on a liquid crystal display (CFAH1604A-YYH-JT, Crystalfontz). In experiments, both the output from the three-channel receiver and the raw data from the detector were available, with the raw data being used to analyse the robustness of the signal detection, particularly in the case of two-threshold error correction (see below). Although only one bit of information is extracted from the three channels (corresponding to each wavelength) using this setup, one could encode more information into the intensity levels of each channel, thereby increasing the number of states. The detection system can be adapted to discern multiple intensity levels, either for higher density encoding or to implement the two-threshold correction system below by adding one more TLC3704 chip to the system.

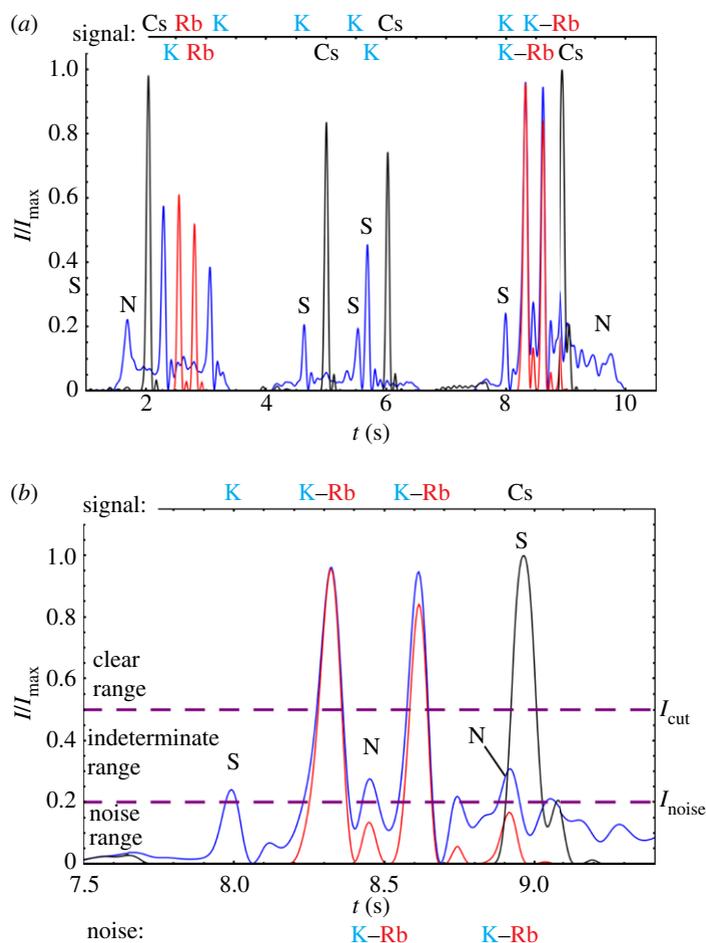


Figure 2. Errors in the infofuse. Salient intended peaks are labelled with an ‘S’, and noise peaks with an ‘N’. (a) A sample of the errors occurring for the infofuse. Noise peaks at the beginning and at the end of transmission have a somewhat broader profile in this experiment, but many intended peak intensities well above background. The intensity of the noise near 2 s is of the order of the intended peaks near 4.5–5.5 s. Two peaks occur near 5.5 s, with a temporal spacing smaller than expected. (b) Intended peaks may be difficult to distinguish between noise peaks in both maximal intensity and intensity profile. The noise peak near 9 s could be considered either an insertion (noise followed by intended) or permutation (noise and intended giving a K–Rb–Cs signal). (Online version in colour.)

(b) Errors in the infofuse

In figure 2a, we show a typical sample of the measured intensity of the infofuse associated with the transmission of the message (in this case, the encoding for ‘tufts’, described further below) as a function of time, with many of the intended peaks being sharp, well defined (although of variable intensity) and well above background. However, at the beginning and end of this particular transmission, the noise is much larger (typically the higher noise initially may be owing to the

ignition of a match); indeed, decreasing the thresholding level would likely add spurious additional potassium peaks to the beginning and end of the signal and produce insertion errors in the signal. In figure 2*b*, corresponding to the message we see that the intended peak near 8 s (marked as intended in the figure) has a temporal profile similar to the other intended peaks, but with a significantly reduced maximum intensity. This is probably caused either by a misapplication of the metallic spot (too little potassium), or because portions of the pattern did not successfully ignite. The intensities of the noise peaks near 8.5 and 9 s are similar in both intensity and profile to the low intended peak at 8 s, making it difficult to distinguish between signal and noise. The probable cause of these noise peaks are non-uniformity in the flame front or inhomogeneities in the fuse, both of which could delay the ignition of a portion of the applied salt. We note that increasing the concentration of the salt is expected to have little effect on these inhomogeneities, and thus would not likely alter the noise observed at 8.5 or 9 s in figure 2*b*. Improvements in the accuracy and uniformity of the applied solvent would likely partially prevent the drastic reduction in the intensity of intended peaks (as was the case near 8 s in figure 2*b*), but cannot prevent inhomogeneity in emission time.

The three peaks near 8.5 s (occurring at 8.3, 8.45 and 8.6 s) have a very small temporal separation compared with the separation between intended peaks (0.15 and 0.3 s, respectively), and one could reasonably discard the noise peak at 8.45 s based on its proximity to the other peaks. However, if the spacing between spots on the fuse was reduced, then such a large temporal threshold would run the risk of discarding intended peaks as well. The noise peak near 9 s could be discarded (as well below background), treating it as an insertion error (a K–Rb peak followed by a temporally very close Cs peak), or merge with the intended peak (with the intended Cs peak read as a K–Rb–Cs peak). While the particular values of the thresholds in temporal spacing and intensity will have an effect on the noisiness of the signal sent by the infofuse, experimentally, we find that the primary sources of error are associated with the insertion of unintended bits and possible permutations of intended bits. Importantly, we do not have the problem of any *missing* peaks over the range of operation of our receivers, so that we do not need to worry about deletion errors as we develop a robust error-correction scheme for the infofuse.

In order to overcome the noise in transmission, a simple solution is to increase the spacing between the metallic patterns. This will cause all the noisy peaks to be well separated, so that unambiguously determining the applied chemicals is virtually assured without confusion owing to neighbouring peaks. However, this will not only decrease the physical rate of the information transfer, but also the length of the signals that can be sent (as messages must be encoded on a fuse of finite length). If we have n intended peaks separated by a distance d_0 , and the average flame front velocity is v , then the physical rate is $r_0 = v/d_0$, with a time approximately n/r_0 required to transmit the message. By increasing the spacing from d_0 to d , we decrease the physical rate of order nd_0/d . In figure 2, the temporal separation between intended peaks is of the order of the width of the chemical patterning. Increasing the spacing by a factor of 2 will significantly separate the peaks, simplifying the process of distinguishing between data and noise, but at the cost of cutting the physical rate in half and doubling the length of the transmission time.

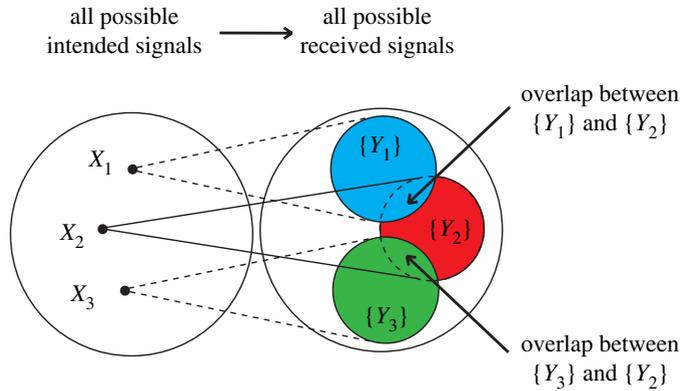


Figure 3. Conceptual diagram of error-correcting codes (Shannon 1948; Hamming 1950). Each intended signal X_i that can be sent will map onto a set of received signals $X_i \rightarrow \{Y_i\}$ that differ from the original. By restricting the sent messages to those whose mappings $\{Y_i\}$ do not overlap for all i , the intended signal can be perfectly reconstructed. If the overlap is small, then the correct signal can only be recovered with high probability. (Online version in colour.)

An alternative to simply increasing the distance between peaks is to introduce a self-correcting code into our encoding for the infuse, which will allow the correct signal to be reconstructed in the presence of noise. Such a code will be preferable to simply increasing the distance between peaks if it is more efficient (i.e. allows a message to be reliably sent at a higher rate). A variety of error-correcting codes have been developed to allow the recovery of a noisy signal under differing noise conditions. In a perfect world, an error-correcting code would be designed so that any code word sent could not be confused for any other possible code word, regardless of the errors that occur (figure 3). The classic Hamming (Hamming 1950; Cover & Thomas 1991) or Golay (Cohen *et al.* 1997) codes, the more commonly used Reed–Solomon codes (Wicker & Bhargava 1994) or the more modern and extremely high rate low-density parity check (LDPC) codes (Gallager 1962; MacKay 1999) allow the correction of up to a fixed number of permutation errors $\lfloor (h-1)/2 \rfloor$ (with h the minimum or Hamming distance of the code), with 100 per cent probability. All of these codes are effectively implemented for the correction of permutation errors, where the intended bit value d_i may be permuted via channel noise into an unintended value d'_i . However, experimentally, we find that the infuse suffers from insertion errors (where bits may be added to the signal; figure 2), which cannot be handled using these well-known codes. While error-correcting codes that address insertion or deletion errors have been studied (Tenengolts 1984; Klove 1995; Davey & MacKay 2001; Drinea & Mitzenmacher 2007; Mitzenmacher 2009) from a variety of approaches, many are not optimally adaptable to non-binary codes. LDPC codes, which are extremely efficient for long signals, are somewhat inappropriate for the short signals that must be sent via the infuse, owing to the physical constraints on the length of the fuse. Additionally, codes that are capable of correcting insertions *and* deletions, while only insertions are observed experimentally, are expected to be less efficient than a code that focuses solely on insertion errors.

3. Coding scheme

(a) Check bits

We wish to develop a simple scheme that uses the non-binary nature of the infofuse to correct an arbitrary number of errors with high probability (MacKay 1999). As seen in figure 2a, if we choose a single, sufficiently low-intensity threshold, we can be certain of correctly including all data peaks, but must accept that some noise peaks will be inserted. Alternatively, if we choose a pair of thresholds with one sufficiently high, we can be sure that all noise peaks are excluded from the signal, but some data peaks may be excluded as well. These dropped data peaks will join a set of *indeterminate* peaks: intensity peaks which clearly indicate that there was some chemical present (well above background), but whose intensities are not high enough to be deemed ‘clear’ data peaks. In this case, the receiver must be able to accurately determine which of the indeterminate peaks were intended, and which are noise.

To allow error correction in the infofuse, the sender and receiver must agree on the number of data bits, n , and the number of check bits, m , beforehand. The check bits are chosen to convolve both the position and the value of each data bit in a unique way, with the first $N - 1$ bits of the form

$$c_k = \sum_i i^{k-1} d_i \bmod N, \quad (3.1)$$

where d_i is the data bit value at the i th position and $N = 7$ is the number of possible bit values. This simple form for the check bits is suboptimal in many respects, but has the merits of clarity, simplicity and flexibility (see electronic supplementary material for further discussion). It is not difficult to see that the probability of a randomly chosen sequence producing a given set of $\{c_k\}$ is $p_{\text{fail}} = N^{-m}$ (similar to the discussion in §8 of Mitzenmacher 2009). This exponentially decaying probability will allow the determination of a robust error-correcting code. Below, we describe the correction scheme for insertion errors (see electronic supplementary material for discussion of permutation errors). While codes that have the ability to correct a single insertion or deletion error with 100 per cent probability convolve data and position with two constraints (Tenengolts 1984; Sloane 2002), we will see the use of multiple check bits and large-alphabet ($N = 7$) encoding allows us to correct an arbitrary number of insertion errors with high probability. We note that the advantages of non-binary communication do not depend on the use of an infofuse in particular, but could be applied to any communication system with many states per bit.

(b) Single-threshold correction

If a single-intensity threshold is used (figure 2), the received signal will be a combination of all $n + m$ data and check peaks, as well as $k \geq 0$ noise peak insertions. Our error-correction method will fail if a bit is truly deleted, but by choosing a sufficiently low threshold, we are able to ensure that all intended peaks are accurately recovered. However, this restriction is counterbalanced by the efficiency of our code (see below) that does not require the need to correct errors where an intended bit is deleted. For the moment, we assume the m check

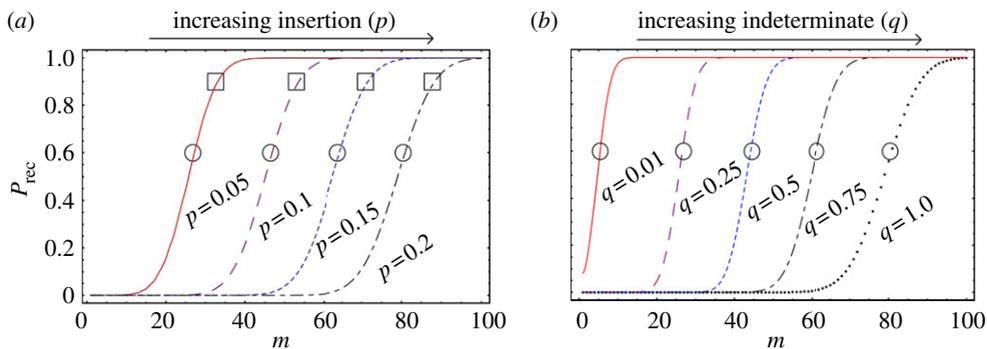


Figure 4. Recovery probability for insertion errors. (a) Shows P_{rec} as a function of the number of check bits m for varying insertion probability p , all with $n = 250$ (solid, $p = 0.05$, dashed 0.1, dotted 0.15 and dashed-dotted 0.2). The large circles are centred on the asymptotic value of $m = nH_N(p)/(1 - p)$, whereas the large squares are centred on the higher order solution in equation (3.3). (b) Shows the recovery probability using a pair of thresholds with $p = 0.2$ and $n = 250$ for varying m and q (q being the probability of an intended peak found below the clear threshold). Shown are $q = 0.01$ (solid line), 0.1 (dashed line), 0.25 (dotted line), 0.5 (dashed-dotted line) and 1 (black points). $q = 1$ is identical to the $p = 0.2$ curve in (a). Large circles denote the predicted midpoint value m_0 in equation (3.5). (Online version in colour.)

bits are perfectly recovered, an unrealistic and severe approximation that will be discussed in detail below. The possibility of permutation errors is also neglected here (see electronic supplementary material for further discussion). In order to model the noisy transmission, we assume a uniform probability p that a noise peak is inserted following any observed peak in the data block. The number of insertion errors observed in the system then has the distribution $P_{\text{ins}}(k) = \binom{n+k-1}{k} p^k (1-p)^n$, giving $\langle k \rangle = np/(1-p)$, with a variance $\langle k^2 \rangle - \langle k \rangle^2 = np/(1-p)^2$. A receiver who finds a signal with $n+k$ peaks can simply determine all possible subsequences of length n and compare the received check bits with the computed values (where the final m bits are assumed correct, see below for further discussion). There will be $\binom{n+k}{n}$ such sequences, so long sequences with multiple errors have an extremely high computational cost. The probability of recovering only the correct signal given k insertion errors is $P_{\text{rec}}(k) \gtrsim (1 - N^{-m}) \binom{n+k}{n}^{-1}$, and the total recovery probability (i.e. the probability that a single, unique sequence is recovered) is

$$P_{\text{rec}} \gtrsim \sum_{k=0}^{\infty} P_{\text{ins}}(k) (1 - N^{-m}) \binom{n+k}{n}^{-1}. \quad (3.2)$$

The recovery probability as a function of m is shown in figure 4, and it is clear that P_{rec} is sigmoidal in nature and increases rapidly beyond the midpoint of the transition. Equation (3.2) incorporates the total number of possible trial sequences rather than unique trial sequences (neglecting any correlations between the trial sequences), and thus will underestimate the probability of recovery.

A sequence containing the average number of insertion errors, $k = \langle k \rangle = np/(1-p)$, will require $\binom{n+k}{n} \sim \exp[nH_e(p)/(1-p)]$ trial sequences, where $H_e(p) = -p \ln(p) - (1-p) \ln(1-p)$, similar to the binary entropy function (Cover & Thomas 1991). Because of the exponential growth of the computational complexity of the code, in practical implementations, we must choose the number of data bits n , such that the number of expected trial sequences is not too large. We expect the number of errors to scale as $k_0 \approx \langle k \rangle = np/(1-p)$, with higher order corrections scaling as $\delta k_0 \propto \sqrt{\langle k^2 \rangle - \langle k \rangle^2} \sim \sqrt{np}/(1-p)$, where the proportionality constant is of order unity. It is not difficult to see that $P_{\text{rec}}(k_0 + \delta k_0) \sim 1 - \epsilon$ when $m = m_0 \sim -\log_N[1 - (1 - \epsilon)^{\binom{n+k_0+\delta k_0}{n}}]$, yielding

$$\left. \begin{aligned} m_0 \sim n \frac{H_N(p)}{1-p} - \sqrt{n} \frac{\log_N(p) \sqrt{p}}{1-p} + g(n, \epsilon) \\ \text{and} \\ H_N(p) = -p \log_N(p) - (1-p) \log_N(1-p), \end{aligned} \right\} \quad (3.3)$$

with $g(n, \epsilon)$ an undetermined function satisfying $g(n, \epsilon)/\sqrt{n} \rightarrow 0$ as $n \rightarrow \infty$. g can in principle be determined numerically, but is analytically difficult to determine directly. This argument determines the number of check bits required to recover from $k_0 + \delta k_0$ errors with high probability, and essentially estimates the number of errors that may occur while the overlap in output sequences remains small (as shown in figure 3). The asymptotic values of m are shown in figure 4a for various values of p , and the leading order term in n roughly coincides with the midpoint of the transition between low and high recovery probability. As $n \rightarrow \infty$, the dominant contribution to m will be the number of check bits required to reach the transition.

(c) Higher rates using multiple thresholds

Improved recovery statistics can be attained by using a pair of thresholds, dividing the signal into clear, indeterminate and background ranges. Peaks in the indeterminate range will be a mix of intended peaks and noise peaks, and the error-correction scheme must be able to distinguish between the two. As was the case for the single threshold, it is essential that the noise threshold is low enough that we can be certain that all intended peaks are detected. Likewise, we assume the threshold I_{cut} is chosen sufficiently high so that no noise peaks ever fall into the clear range, else our coding scheme would fail. The receiver will find $n-l$ clear peaks, as well as $k+l$ indeterminate peaks (with l the number of intended peaks that are considered indeterminate and k the number of insertions). In order to recover the intended signal, all possible sequences of length n containing all $n-l$ clear peaks and l of the indeterminate peaks can be generated, and compared with the check bits. Again, this decoding scheme has high computational complexity for long, noisy signals, with $\binom{k+l}{l}$ trial sequences being generated. However, the number of trial sequences using a pair of thresholds is strictly less than that when a single threshold is used.

We assume that the probability of an intended peak being considered indeterminate is uniform with probability q , and maintain the stringent

assumption that all m check bits are perfectly recovered. The probability of recovering the intended signal uniquely is then

$$P_{\text{rec}} = \sum_{k=0}^{\infty} \sum_{l=0}^n P_{\text{ins}}(k) P_{\text{del}}(l) (1 - N^{-m})^{\binom{k+l}{l}-1}, \quad (3.4)$$

with $P_{\text{del}}(l) = \binom{n}{l} q^l (1-q)^{n-l}$ and $P_{\text{ins}}(k)$ the uniform probability of insertion. Equation (3.4) counts all possible trial sequences, not just unique trial sequences, and thus underestimates the probability of recovery. The average number of indeterminate peaks will be $\langle k+l \rangle = nq + np/(1-p)$, and we can perform the same analysis as in the single-threshold case to estimate the number of bits required,

$$m_0 \sim \frac{n}{1-p} \left[(p+q-pq) \log_N \left(\frac{p}{1-p} + q \right) - p \log_N \left(\frac{p}{1-p} \right) - (q-pq) \log_N(q) \right] + O(n^{1/2}). \quad (3.5)$$

Equation (3.5) is equivalent to equation (3.3) if the probability of dropping an intended peak $q=1$, and decreases with decreasing q . In figure 4*b*, we see that for fixed insertion probability p , the recovery probability has a sharper transition for far lower m as q decreases (see electronic supplementary material, figure 1 as well). For decreasing q , the indeterminate error correction becomes more reliable than the insertion correction. A pair of thresholds not only decreases the computational complexity, but also greatly increases the reliability of communication.

(d) Meta-check bits

In our determination of the number of check bits required to correctly recover the intended sequence, we have thus far made the stringent assumption that the check bits are perfectly recovered. The results presented above can be used with minimal modification if we can be sure of recovering the check bits with high probability. This immediately suggests the use of meta-check bits, which perform the same redundancy on the check bits that the check bits perform on the data. The meta bits will have the form $c'_k = \sum_i i^{k-1} c_i \bmod N$, much similar to equation (3.1). Each of these meta-check bits can suffer from insertion or indeterminate errors as well, so an additional set of bits must be used to check all meta bits as well. The multi-layered level of protection can be continued indefinitely (discussed further in the electronic supplementary material), allowing for an iterative protection scheme ensuring reliable recovery.

In the limit of large n , the k th meta-check block will require $m_k/n = (m_0/n) \times (m_{k-1}/n)$ bits (with m_0 given in equations (3.3) or (3.5)), i.e.

$$m \sim n \sum_{k=1}^{\infty} \left(\frac{m_0}{n} \right)^k = n \frac{m_0}{n - m_0}, \quad (3.6)$$

with the rates $R = n/(n+m) = 1 - m_0/n$ of these codes shown in figure 5. The rate vanishes at a finite value of p for all q , owing to the fact that a sufficiently noisy transmission would require $m_0 > n$. At worst (with $q=1$ in

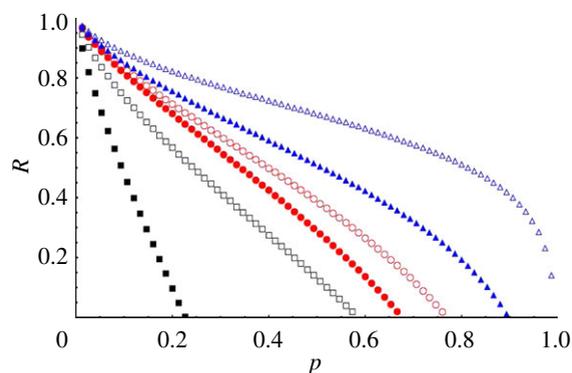


Figure 5. The predicted information rate using meta-check bits from equation (3.6). Shown is the rate in the indeterminate channel with $q = 0.25$ (empty triangles), 0.5 (filled triangles), 0.75 (empty circles) and 1.0 (filled circles) for $N = 7$. $q = 1.0$ is equivalent to the single-threshold corrector. The filled black squares show the rate for $q = 1$, but for the binary $N = 2$, and the empty black squares show the rate for the trinary encoding with a division of bits into bytes of length $b = 2$ ($N_{\text{byte}} = 9$). Both these clearly display the increased efficiency of large alphabet encoding. (Online version in colour.)

the single-threshold case), the rate vanishes at $p_{\text{max}} \approx 0.68$. Interestingly, for a binary channel with $N = 2$, we find $p_{\text{max}} \approx 0.22$, showing the increased efficiency owing to the non-binary coding. The indeterminate error-correction scheme has an even higher rate as q (the probability of an intended peak being considered indeterminate) decreases.

It is worth noting that an alternative to our large alphabet encoding ($N = 7$) could be replaced with a binary or trinary system of individual peaks (i.e. $K = 0$, $C_s = 1$ and $R_b = 2$), which are grouped together in bytes of length b . Using three elements, each byte can attain $N_{\text{byte}} = 3^b$ states, so it is useful to determine the efficiency of our error-correcting code using such an encoding system. A system that uses $b = 2$ (i.e. $N_{\text{byte}} = 9$) would still require a pair of bytes to represent an alphanumeric character, but would require twice as many peaks as the $N = 7$ case to send the message. Equation (3.3) can be adapted in a straightforward fashion, and we find that $p_{\text{max}} \approx 0.59 < 0.68$ for the byte-wise trinary signal. Such a method is thus less efficient than the large alphabet encoding used, despite the fact that $N_{\text{byte}} = 9 > N = 7$, both in the expected rate of the code (figure 5), as well as in the required fuse length to send a message ($b = 2$ transmission would require twice the length of fuse to send the same signal). Bytes grouped with $b \geq 3$ do allow a higher information rate than the large alphabet $N = 7$ encoding, but are more inefficient in terms of fuse length (with a $b = 3$ encoding having $p_{\text{max}} = 1$ but requiring 1.5 times the number of bits to send the message). The large alphabet encoding presented here thus strikes an excellent balance between length efficiency and error-correction efficiency, important in the context of infochemistry.

4. Experimental results

In order to test the applicability of our coding scheme to the experimental system, we sent the signal ‘tufts’ 19 times using three check bits and one meta-check bit (14 total bits, with a rate $n/(n + m) = 0.71$, the encoding scheme is presented in

the electronic supplementary material). The intended signal is 10220010010441, and translates to ‘tuftsiz’ with the four additional error-correcting bits. For each experiment, the intensity of each channel (K, Rb and Cs) was rescaled by the maximum intensity. Each message was decoded from the raw data with a noise threshold of at least $I_{\text{noise}} = 0.12I_{\text{max}}$ for all experiments. However, if more than five insertion errors were detected (the received signal comprised more than 19 bits), the noise threshold was increased to $I_{\text{noise}} = 0.15I_{\text{max}}$ to reduce the number of detected errors. In most cases, the threshold of $I_{\text{noise}} = 0.12I_{\text{max}}$ was unnecessarily low, but there were some experiments where the intensity of an intended bit was of the order of this threshold).

This threshold was applied to all experiments to be absolutely certain that all data peaks would be included. A higher threshold would have produced fewer channel errors by ignoring more of the inserted peaks, so error correction with this threshold demonstrates a worst-case scenario. Peaks were considered to occur simultaneously if their maxima were within 0.08 s, and peaks separated by more than 0.11 s from both of their nearest neighbours would be included in the returned sequence (with all other peaks discarded as noise). The temporal thresholding is also lower than is strictly necessary, as the average spacing between intended peaks is approximately 0.3 s. The use of a lower threshold again ensures that no intended peaks are dropped at the cost of additional insertions or permutations. Using these thresholds with the time trace shown in figure 2, we recover a signal with five insertions and one permutation error: 0102200100104040063. Using a pair of thresholds, we recover the signal **010220010010404036**, where bold-faced numbers correspond to intensities above I_{cut} , and therefore are assumed intended (but possibly permuted), and the other numbers are peaks with intensities between I_{cut} and I_{noise} .

There are a variety of improvements that can be made in the signal processing to reduce the number of errors in the received signal. Although not implemented here, improvements can be made on the detection of clear and indeterminate peaks by using energy-averaging or matched filtering of the signal (J. Kusuma 2011, personal communication). Many spurious peaks are detected by searching for local maxima in the intensity profiles (the method we use in this paper), only some of which can be discarded as spurious using a temporal threshold. By integrating the intensity over M time windows $\mathbf{J}_k = \int_{t_0+kT}^{t_0+(k+1)T} \mathbf{I}(t)$ (where $\mathbf{I} = (I_K, I_{\text{Cs}}, I_{\text{Rb}})$ is the vector of intensities), the spurious local maxima will have a reduced impact on the final digitized signal. The correct values of T and t_0 must be selected via an optimization procedure for each fuse, as small variations in the flame front velocity can alter T . We note that if two intended peaks are found within the same time interval T using the integrated signal, a deletion error would have occurred in the received signal (as the maximum number of peaks is M). Our error-correcting code will fail if such a deletion error occurs. The advantages and disadvantages of alternate methods of digitizing the signal (perhaps requiring different methods of error correction) are not clear when compared with the approach presented in this paper.

Because of the limited field of vision of the telescopic detector over short ranges, the signal was divided into three fuses, with five, five and four bits, respectively. This led to increased separation between blocks of bits (figure 2*a*). The very limited range of temporal cutoffs (a range of 0.03 s for discarding peaks as noise) suggests that this increased spacing does not significantly alter the

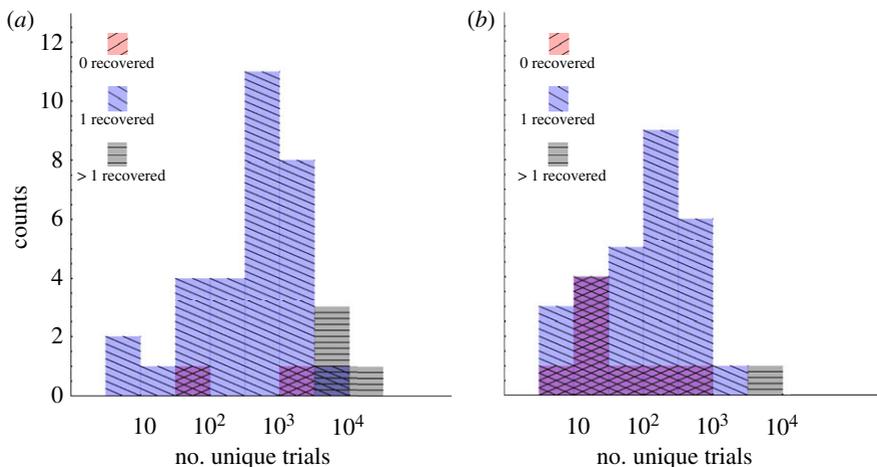


Figure 6. The error correction of the message ‘tufts’, using three check bits and one meta-check bit ($n = 10$, $m_0 = 3$, $m_1 = 1$). (a) Shows the recovery using the $I_{\text{cut}} = 0.6I_{\text{max}}$ and (b) shows the recovery using $I_{\text{cut}} = I_{\text{max}}$. Both include the results correcting for 0, 1 and 3 permutation errors. Upward slashes denote failed recovery, downward slashes denote unique recovery and horizontal slashes denote multiple-recovered sequences. Shown are only those sequences that gave fewer than 2×10^4 unique trials, which excludes seven (14%) of the attempts from (b). (Online version in colour.)

noise statistics. When the two thresholds were used, the threshold for clear peaks was set to $I_{\text{cut}} = 0.5I_{\text{max}}$ (figure 6*b*). In all cases, we first tried to correct the signals without regard for permutation errors, but often found that the signal was not accurately recovered. We then corrected for a single permutation error (see electronic supplementary material for discussion of permutation correction), which ensured the recovery of the correct sequence in all experiments.

In figure 6, we show histograms of the results of the error correction, using a single (figure 6*a*) or pair of thresholds (figure 6*b*). In both, we show the results when correcting for both zero and one permutation error in the same histogram. We divide the experimental results into attempted recoveries for which no trial sequences satisfied the check bits (upward slashes), those that recovered a unique sequence (downward slashes) and those for which multiple sequences were recovered (horizontal slashes). This histogram is plotted as a function of the number of unique trial sequences generated, rather than the $\binom{n+k}{n}$ sequences expected from the random coding arguments above. The average results are also listed in table 1. We note that in all but one case, if any signals were recovered one of them was the intended ‘tufts’; however, other spurious matches were possible. In all cases, where the unique sequence was not recovered, a permutation error had occurred but was uncorrected. In general, the number of unique trial sequences generated using a pair of thresholds was lower than the number using when a single threshold was used. This is owing to the additional information created by labelling some peaks as ‘certainly’ intended, and greatly reduces the computational complexity of decoding.

In figure 6, all occurrences of zero-recovered sequences corresponded to at least one permutation error occurring. Correcting permutation errors can greatly increase the number of trial sequences that must be sampled, and increases the

Table 1. Summary of the recovery statistics using one or two thresholds. The first column shows the number of permutation errors corrected, the second column the geometric mean of the number of unique trials generated, the third the fraction of trials that found the correct sequence and the fourth the average number of returned sequences. Without correcting for permutation errors, the correct sequence was recovered at best 58% of the time, but false positives were very rare ($\langle \text{correct} \rangle \approx \langle \text{recovered} \rangle$). The correct message was recovered for all experiments if one permutation error was corrected, and slightly increased the number of false positives ($\langle \text{recovered} \rangle > 1$).

permutations	$\overline{\text{trials}}$	$\langle \text{correct} \rangle$	$\langle \text{recovered} \rangle$
recovery statistics with two thresholds			
0	33.9	0.58	0.58
1	202	1.0	1.16
recovery statistics with one threshold			
0	192	0.84	0.89
1	883	1.0	1.53

probability of finding more than one recovered sequence. The unique intended signal ‘tufts’ was recovered as long as the number of unique trial sequences was less than 3418 (giving rise to three matches), while at worst five sequences were recovered (the intended and four spurious matches) for 21 015 unique trials. The largest number of trial sequences that resulted in a unique (and correct) match was 4207. This shows the importance of correlations in the trial sequences, as for a randomly drawn set of trials, we would expect the probability of recovering a unique match as $P_{\text{rec}} < (1 - 7^{-3})^{4207} \approx 5 \times 10^{-6}$ (see equation (3.2)). The random coding argument presented above significantly underestimates the probability of recovery, and the surprisingly small number of recovered sequences given the number of trial sequences shows the importance of correlations between trial sequences. It is worthwhile to note that even in cases where multiple sequences satisfy all of the check bits, the error-correction scheme still produces a drastic reduction in the number of possibly intended messages. In the worst case, with 21 000 trial sequences, the five recovered messages translate to ‘tufts’, ‘saets’, ‘saosr’, ‘tess3’ and ‘_efs*’. While it is clear that the possibility of multiple spurious matches is a limitation of our coding scheme, it is equally clear that the 21 000 possible sequences are reduced to a set of five, for which the english message is clearly distinguishable.

5. Conclusions

We have presented a relatively simple method for error correction for messages sent via a burning fuse patterned with metallic salts. Our coding scheme depends on the experimentally observed noise properties of the system: while noise peaks may cause insertion or indeterminate errors (by being indistinguishable from data peaks), no information is truly lost from the system. Even with the simplest possible representation for the redundant error-correcting bits (see electronic supplementary material), the experiments show that our correction scheme is able to recover the intended signal in the presence of noise with high probability.

By introducing layering in the error correction, in the form of the meta-check bits, it is possible to achieve arbitrarily high probability of signal recovery, assuming that errors are sufficiently rare. A random-coding argument shows that there is an achievable, finite rate for error correction in an idealized infofuse system. However, the experiments show a much higher probability of recovery than would be expected given the random coding argument, owing to the correlations in trial sequences.

The highly efficient nature of the code is due primarily to the non-binary nature of the encoding in the infofuse. A binary or trinary signal can be encoded by grouping bits into bytes of finite length in order to produce a large number of states per byte, but has an associated cost in increasing the required length of the signal. We have seen that not only does byte-wise trinary communication require effectively doubling the length of the signal, but also that the increase in the average number of errors reduces the efficiency of the code in comparison with a large N bit-wise coding. It is worthwhile to note that this code could be applied to a system where a larger N is used for encoding, such as increasing the number of emissive salts used in the preparation of the infofuse. Likewise, higher information density could be achieved by using the concentration of salts in each spot to encode information (Thomas *et al.* 2009). While the choice of appropriate thresholds could be more difficult in this situation, we expect that increasing the alphabet size will give an increase in the efficiency in the code (Mitzenmacher 2006).

The particular nature of the noise observed in the infofuse, coupled with the great advantages in large N encoding, suggests that there may be great advantages in certain cases in chemical communication. One can envision a multitude of physical or chemical systems in which the number of ‘bits’ can represent many states (large N) in a novel form of communication. For each, all that remains is a more complete understanding of the errors caused in transmission, and how to design an error-correcting code that takes advantage of the noise characteristics for each. In some cases, the many well-developed codes designed for binary communication may be adaptable or immediately applicable, but new techniques may be required to fully use the advantages of each system while simultaneously overcoming the inherent disadvantages in the same.

We gratefully acknowledge useful conversations with J. Kusuma regarding methods of digitizing the signal.

References

- Cohen, G., Honkala, I., Litsyn, S. & Lobstein, A. 1997 *Covering codes*. Amsterdam, The Netherlands: Elsevier Science BV.
- Cover, T. & Thomas, J. 1991 *Elements of information theory*. New York, NY: Wiley.
- Davey, M. C. & MacKay, D. J. C. 2001 Reliable communication over channels with insertions, deletions, and substitutions. *IEEE Trans. Inf. Theory* **47**, 687–698. (doi:10.1109/18.910582)
- Drinea, E. & Mitzenmacher, M. 2007 Improved lower bounds for the capacity of i.i.d. deletion and duplication channels. *IEEE Trans. Inf. Theory* **53**, 2693–2714. (doi:10.1109/TIT.2007.901221)
- Gallager, R. G. 1962 Low-density parity-check codes. *IEEE Trans. Inf. Theory* **8**, 21–28. (doi:10.1109/TIT.1962.1057683)
- Hamming, R. W. 1950 Error detecting and error correcting codes. *Bell System Tech. J.* **29**, 147.
- Hashimoto, M. *et al.* 2009 Infochemistry: encoding information as optical pulses using droplets in a microfluidic device. *J. Am. Chem. Soc.* **131**, 12420–12429. (doi:10.1021/ja904788m)

- Kim, C., Thomas, S. W. & Whitesides, G. M. 2010 Long-duration transmission of information with infofuses. *Angew. Chem.* **122**, 4675–4679. (doi:10.1002/ange.201001582)
- Klove, T. 1995 Codes correcting a single insertion/deletion of a zero or a single peak-shift. *IEEE Trans. Inf. Theory* **41**, 279–283. (doi:10.1109/18.370100)
- MacKay, D. J. C. 1999 Good error-correcting codes based on very sparse matrices. *IEEE Trans. Inf. Theory* **45**, 399–431. (doi:10.1109/18.748992)
- MacKay, D. J. C. 2003 *Information theory, inference, and learning algorithms*. Cambridge, UK: Cambridge University Press.
- Mitzenmacher, M. 2006 Polynomial time low-density parity-check codes with rates very close to the capacity of the q -ary random deletion channel for large q . *IEEE Trans. Inf. Theory* **52**, 5496–5501. (doi:10.1109/TIT.2006.885443)
- Mitzenmacher, M. 2009 A survey of results for deletion channels and related synchronization channels. *Prob. Surv.* **6**, 1–33. (doi:10.1214/08-PS141)
- Peterson, W. W. & Weldon, E. J. 1972 *Error-correcting codes*, 2nd edn. Cambridge, MA: MIT Press.
- Ralchenko, Y., Kramida, A. E., Reader, J. & NIST ASD Team 2010 NIST atomic spectra database (version 4.0). Gaithersburg, MD: National Institute of Standards and Technology. See <http://physics.nist.gov/asd>.
- Scoog, D. A., Holler, F. J. & Crouch, S. R. 2007 *Principles of instrumental analysis*. Belmont, CA: Thomson Brooks/Cole.
- Shannon, C. E. 1948 A mathematical theory of communication. *Bell System Tech. J.* **27**, 379.
- Sloane, N. J. A. 2002 On single-deletion-correcting codes. In *Codes and designs* (eds K. T. Arasu & A. Seress), pp. 273–291. Berlin, Germany: Walter de Gruyter.
- Tenengolts, G. 1984 Nonbinary codes, correcting single deletion or insertion. *IEEE Trans. Inf. Theory* **30**, 766–769. (doi:10.1109/TIT.1984.1056962)
- Thomas, S. W. *et al.* 2009 Infochemistry and infofuses for the chemical storage and transmission of coded information. *Proc. Natl Acad. Sci. USA* **106**, 9147–9150. (doi:10.1073/pnas.0902476106)
- Wicker, S. B. & Bhargava, V. K. 1994 *Reed–Solomon codes and their applications*. Piscataway, NJ: IEEE Press.

Robust error correction in Infofuses: Supplementary Information.

Greg Morrison, Sam W. Thomas III, Christopher N. LaFratta, Jian Guo,
Manuel A. Palacios, Sameer Sonkusale, David R. Walt,
George M. Whitesides, and L. Mahadevan

March 7, 2011

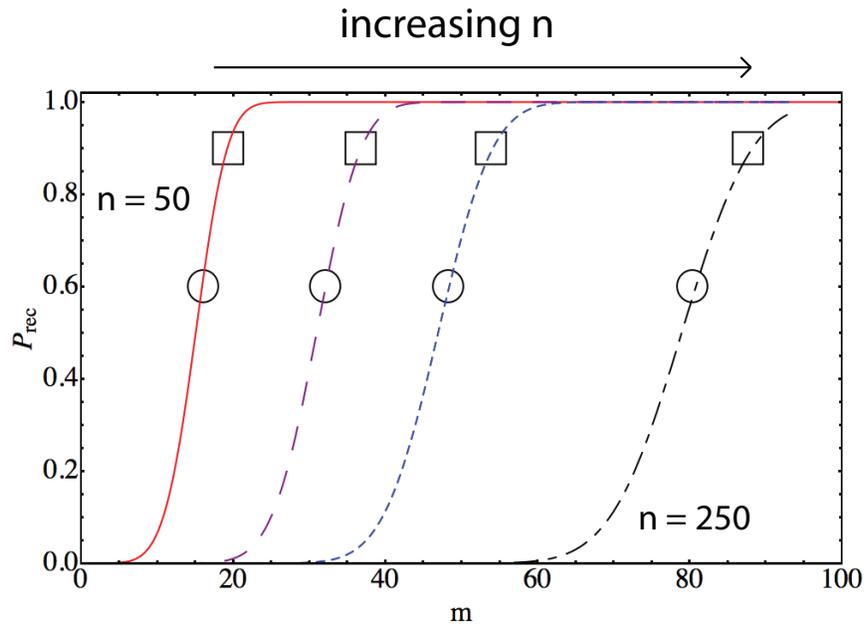


Figure 1: Predicted recovery probability using the single threshold decoding for fixed probability $p = 0.05$ as a function of m for varying n . Solid red shows $n = 50$, dashed purple 100, dotted blue 150, and dash-dotted black 250. The large squares and circles show the predicted number of required check bits, as in Fig. 4 of the main text

1 Arbitrary recovery probability

An arbitrarily high probability of recovery can be demonstrated by choosing some k_M such that the probability of seeing more than k_M errors is $\leq \epsilon_1$ (i.e. $\sum_{k=0}^{k_M} P_{ins}(k) \geq 1 - \epsilon_1$). Further, we choose the number of check bits m such that $P_{rec}(k = k_M) \geq 1 - \epsilon_2$. The probability of recovery is thus

$$P_{rec} = \sum_{k=0}^{\infty} P_{ins}(k)P_{rec}(k) \geq (1 - \epsilon_2) \sum_{k=0}^{k_M} P_{ins}(k) \geq (1 - \epsilon_1)(1 - \epsilon_2) \quad (1)$$

which can be made arbitrarily close to 1.

2 Optimization of the form of the check bits

The form of the check bits in eq. 1 of the main text (and likewise the form of the meta-check bits) is suboptimal, for a number of reasons. First, and perhaps most significantly, each bit at position $i \times N$ is protected *only* by the first check bit, rather than all check bits, because $iN \bmod (N) = 0$. A simple improvement would be to replace the multiplying coefficient i^{k-1} with $(i^*)^{k-1}$ where $i^* - 1 = i \bmod (N - 1)$. i^* is chosen such that none of these coefficients is divisible by N . We are greatly aided by the fact that $N = 7$ is a prime; for more general N we would require each of the coefficients to be relatively prime to N . An additional limitation of the form of the check bits in eq. 1 is that only N independent check bits can have this form. The 8th check bit will have the form $c_8 = \sum_i i^7 d_i \pmod{7} = \sum_i i d_i \pmod{7} = c_1$, since $i^7 = i \pmod{7}$ from Fermat's little theorem [1]. c_7 is highly correlated with c_0 , differing only at the values of data points occurring at positions $i \times N$. In all of our experiments, we did not consider more than 4 check bits, so this concern is not relevant. To reliably send longer sequences (requiring more check bits), this is a significant issue, and Eq. 1 from the main text must be modified. A few examples could be $c_{k,l} = \sum_i (i^*)^k d_i^l \pmod{N}$ for $l \leq N - 1$, or $c_{k,l} = \sum_i (i^*)^k d_i d_{i+l}$ for $l < n$. The latter is valid as long as the number of check bits is less than n .

For an arbitrarily long sequence, error correction in the form of a low-density parity check (LDPC) code may be optimal, where all of the data is constrained by parity checks in a sparse bipartite graph[2]. Extensive studies of LDPCs have shown them to be extremely efficient[3], coming very close to the theoretical maximum in a number of cases. LDPCs were not chosen here because they are most appropriate for long sequences, whereas communication using the infofuse is constrained by the length of the fuse. In the case of long sequences, Mitzenmacher has shown that large alphabet encoding can be exploited in the framework of an LDPC to give high-rate codes[4].

3 Permutation Errors

Regardless of the threshold chosen, an unexpected overlap between two peaks may give rise to a permutation error. Permutation errors in the infofuse are unique because no information is truly lost in the system. A Li peak will never be mistaken for a Cs peak, but it may be mistaken for a Li/Cs peak. A receiver who finds a pair or triple of simultaneous peaks will consider them *permutable*, in that there may have been a permutation error that produced them. In order to correct a permutation error, the receiver can generate trial sequences by splitting simultaneous peaks into their constituent parts (e. g. the Li/Cs peak would produce three trial sequences, with Li, Cs, and Li/Cs peaks) Because permutable bits may be included in the intended sequence, the probability of recovering a unique signal due to permutation errors is strongly sequence dependent.

When correcting permutation errors, the receiver generates all possible trial sequences formed by the components of each permutable peak. This gives $3^{a_0+a_1b_0+b}$ possible trial sequences, where a_0 and b_0 are the number of intended pairs and triples, respectively, and a and b are the number of pairs and triples due to permutation or insertion errors in transmission. The number of trial sequences grows very rapidly with the number of permutable bits, so that correcting permutation errors using the check bits of the form in eq. 1 in the main text rapidly becomes unwieldy. It may be preferable to adapt the more computationally efficient permutation correcting codes such as Hamming[5, 6] or Reed Solomon[7] to correct for permutation errors, while retaining the presented code for correcting insertion errors.

One advantage of our error correcting code is that bursts of permutation errors can be corrected exactly. It is not difficult to see that if a single permutable bit is received (in the absence of insertion or deletion errors), only one trial sequence will reproduce the first check bit. This means our error correcting scheme can exactly correct one permutation error, as long as there are no pairs or triples in the intended sequence. If there are two permutable bits at positions i and j , the first two check bits can be satisfied by exactly one trial sequence *unless* $i \equiv j \pmod{N}$. Likewise, up to 6 permutation errors can be corrected as long as the distance between permutable bits is not exactly divisible by N . Our error correcting scheme is most applicable for a single burst of permutation errors, where the errors occur over a relatively short range. However, as $n \rightarrow \infty$, the probability of finding a unique trial sequence becomes *independent of* m , the number of check bits. Thus, for long sequences, our choice of the check bits (eq. 1 from the main text) is inappropriate. A LDPC coding scheme will drastically improve the ability to correct permutation errors, but with a somewhat more complex coding and decoding scheme. In practice, we use relatively short sequences, and the probability of permutation errors tends to be relatively low. In the experiments, we initially attempt recovery under the assumption that no permutation errors have occurred, and apply a permutation-correction if the recovery fails.

In all cases where the chemicals were correctly applied, no more than 3 permutation errors were observed.

4 Further improvements

a	(0,2)	b	(1,3)	c	(3,0)	d	(1,5)	e	(2,0)	f	(0,0)
g	(1,4)	h	(2,3)	i	(0,4)	j	(6,0)	k	(3,4)	l	(5,1)
m	(3,1)	n	(3,2)	o	(2,1)	p	(5,0)	q	(5,3)	r	(4,0)
s	(0,1)	t	(1,0)	u	(2,2)	v	(0,3)	w	(1,1)	x	(3,5)
y	(0,5)	z	(4,1)	-	(1,2)	.	(5,2)	!	(4,3)	?	(1,6)
,	(2,5)	;	(4,5)	:	(2,4)	'	(0,6)	((3,6))	(6,5)
0	(6,1)	1	(2,6)	2	(6,2)	3	(4,2)	4	(5,4)	5	(6,3)
6	(6,4)	7	(3,3)	8	(5,5)	9	(4,6)	*	(4,4)	&	(5,6)

Table 1: Optimal alphabet minimizing the number of nearest neighbor peaks with identical chemical species. $\sim 25.1\%$ of the peaks in the sample text have 1 nearest neighbor with one element the same, $\sim 0.2\%$ have more than one element the same. Elements in the bottom two rows did not occur in the sample text, and are arbitrarily chosen. The (6,6) state is reserved as an error character.

Because we will in general include a finite number of chemical spots on an infofuse, only some of which can be check bits, we must accept that there may exist multiple trial sequences that pass the error-correction scheme, and to devise a method to choose between them. However, we are free to choose the alphabet we use (the method of mapping english characters into distinct pairings of peaks), which will allow us to select between trial sequences. We note that noise peaks are likely to share at least one chemical element in common with a nearby peak, due to the fact that noise peaks are not mis-applied chemical strips on the fuse, but rather originate from impurities in an intended peak (or the nitrocellulose itself), or due to difficult to predict fluctuations in the flame front. By choosing an alphabet such that typical sequences will have few nearest neighbor peaks with identical chemical species, we can improve our ability to distinguish between the intended sequence and spurious matches. We note that, under certain conditions, it might be more preferable to choose an alphabet that reduces permutation errors, rather than insertion errors. Such an alphabet would minimize the number of peaks that contained multiple chemical species, and would likely have many characters in common with our the insertion-minimizing alphabet.

Using Monte Carlo simulations, we have generated an alphabet that minimizes the number of peaks which share one or more elements with their nearest neighbors. We began with a random alphabet, and iteratively changed the encoding for a single character at each

MC step. The ‘energy’ of a particular alphabet is taken to be proportional to the total number of elements in common between nearest neighbors (different values of the proportionality constant were chosen in different runs to find fast convergence). In order to model correlations in the English language, we chose as a sample text the book of Genesis from the King James bible [8] (which allows us to account for the correlations between letters in English). The resulting alphabet is shown in Table 1, and we find $\sim 25\%$ of nearest neighbor peaks share identical elements. This can be compared to our original alphabet (generated in an ad-hoc fashion), with $\sim 70\%$ of the nearest neighbors sharing identical elements. This alphabet has the additional advantage that peak pairs or triplets are more rare, thus reducing the number of permutable bits in an intended sequence. Incorrect trial sequences that *do* pass the check bits are likely to differ significantly from the intended sequence as long as the number of check bits is sufficiently large, so that incorrect sequences are expected to have a large number of peaks with identical elements.

In practice, we will send short messages using the infofuse, rather than an entire book. It is then important to determine whether or not developing the alphabet in this way will give good results for shorter sequences. We simulate the transmission of a block of 7 characters (14 bits) and 2-4 check bits with an insertion probability of $p = 0.1$ 500 times, and determine the number of trial sequences that have more nearest neighbor peaks with identical species. If we take inserted peaks to have a random data value, we find $\sim 20\%$ of the trial sequences that satisfy the check bits have the same number of nearest neighbor peaks composed of the same chemicals and $\sim 70\%$ have more, regardless of the number of checkbits. If we assume that inserted peaks are correlated, such that the inserted data value matches one of its nearest neighbors, we find $\sim 80\%$ of the accepted trial sequences have more nearest neighbors with the same chemicals.

References

- [1] J. Stillwell. *Elements of Number Theory*. Springer-Verlag, New York, 2003.
- [2] D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.
- [3] D. J. C. MacKay. *IEEE Trans. Info. Theory*, 45:399, 1999.
- [4] M. Mitzenmacher. *IEEE Trans. Info. Theory*, 52:5496, 2006.
- [5] T. Cover and J. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.
- [6] R. W. Hamming. *Bell System Tech. J.*, 29:147, 1950.
- [7] S. B. Wicker and V. K. Bhargava. *Reed-Solomon Codes and their applications*. IEEE Press, Piscataway, NJ, 1994.

[8] Project Gutenberg. <http://www.gutenberg.org/etext/8001>.