EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

The combination of sparse learning and list decoding of subspace codes for error correction in random network coding

Guangzhi Zhang[1,2*†], Shaobin Cai[1,3*†], Dongqiu Zhang[4*†], Wanneng Shu[5], Chunhua Ma[2†], Mu Niu[2†] and Xiangnan Ding[2†]

Abstract

The network coding can spread a single original error over the whole network. The simulation shows that the propagated error mostly all the time pollute just 100% of the received packets at the sink if Hamming distance is adopted. If subspace codes are adopted, usually the propagated error will not pollute 100% of the received packets in the sense of subspace distance. However, it also usually pollutes 90% of received packets which is a high error ratio. Even if the rank code and the subspace code are adopted, these existing schemes based on traditional block codes can correct corrupted errors no more than *C*/2 because of the limitation of the block coding where *C* is the max flow min cut. It is an agent to find a dense error correction method in random network coding. List decoding of subspace codes can correct $\frac{C}{k}$-1 errors where *k* is the size of information. When $\frac{C}{k}$ is big, many errors in the sense of subspace distance can be corrected. However, the solution of list decoding is not unique. John Wright proposed a dense error correction technique based on L1 minimization, which can recover nearly 100% of the corrupted observations. In our proposal, the original packets are coded with John Wright's coding matrix, and then, the coded message is coded again with subspace codes. In the sink, the decoding procedures about list decoding of subspace codes and John Wright's scheme are performed. At last, the unique solution is achieved even though there are dense propagated errors in random network coding.

Keywords: Network coding, Error correction, Propagated error, List decoding, Subspace codes

1 Introduction

1.1 The tough problem about error propagation in network coding

In network coding, its very nature of combining information in the intermediate nodes makes it very susceptible to transmission errors. A single error will be propagated by every node further downstream in networks. This will thus prevent the reconstruction of the file in the sink. There are fruitful works about network error correction coding (NEC) in network coding. However, none of the existing works solved the error spread problem in random network coding. If network coding theory is to be applied the commercial theory to the reality, a new method should be proposed to cope with the error spread in the random network coding. A deep understanding of how the error is spread in random network coding is necessary to tackle this difficult issue.

To understand the error propagation in network coding, we will illustrate the transmission model in network coding. A sketch approach is as follows. $\kappa \in (F_Q)^{k\times 1}$ is the original uncompressible message needed to be sent where $k < C$. C is the max flow min cut, and F_Q is the finite field in the source. In the source of a multicast network, $\kappa$is encoded with a MDS (maximum distance separable) linear block code Ω. The block code Ω is with code length C and information length k and is denoted by (C, k). The maximum distance of code Ω is $d_{\min}$. $\kappa \in (F_Q)^{k\times 1}$ is coded

* Correspondence: 409951652@qq.com
Guangzhi Zhang, Shaobin Cai, Dongqiu Zhang, Chunhua Ma, Mu Niu and Xiangnan Ding contributed equally to this work and should be considered as co-first authors.
†Equal contributors
[1]Computer Science Department, Harbin Engineering University, Harbin 150001, China
[4]School of Education Science, Nanjing Normal University, Nanjing 210023, China
Full list of author information is available at the end of the article

into $X \in (F_Q)^{C \times 1}$ with $X = G^{*}\kappa$. $G \in (F_Q)^{n \times k}$ is the generate matrix of code Ω. X is then sent through the network with the network coding scheme. The transport procedure can be expressed by the formulation $Y = T \cdot X + T_{Z \to Y} \cdot Z$ [1]. Z is the error vector which occurs actually in the network. The length of $Z \in (F_Q)^{t \times 1}$ is the number of links where the error occurs. T and $T_{Z \to Y}$ are the transfer matrix in network of X and Z respectively. $Y \in (F_Q)^{C \times 1}$ is the received messages vector in the sink. To decode κ, we first perform decoding algorithm of network coding scheme. The procedure can be expressed by the formulation $X = T^{-1} \cdot Y - (T^{-1} \cdot T_{Z \to Y}) \cdot Z$. Second, the decoding algorithm of Ω is performed to get κ ultimately. If there are no errors which occur in the network, the vector $(T^{-1} \cdot T_{Z \to Y}) \cdot Z \in (F_Q)^{C \times 1}$ will be zero vector, and this moment, $X = T^{-1} \cdot Y = G \cdot \kappa$. κ is mapped to X based on the codebook of code Ω. Therefore, κ can be decoded successfully based on the decoding algorithm of code Ω. Now, we consider the situation that $(T^{-1} \cdot T_{Z \to Y}) \cdot Z$ is not omitted. We denote the number of nonzero components of $(T^{-1} \cdot T_{Z \to Y}) \cdot Z \in (F_Q)^{C \times 1}$ with t'. If $t' < d_{\min}/2$, based on the block coding theory, we can also decode κ successfully. However, because of the impact of $T^{-1} \cdot T_{Z \to Y}$, $(T^{-1} \cdot T_{Z \to Y}) \cdot Z$ will have nonzero components more than t'. That means $t' > t$, and it may $t' < d_{\min}/2$. In this situation, we can decode κ no longer based on the decoding algorithm if the code is Ω. In the sense of network coding, the error $Z \in (F_Q)^{t \times 1}$ is propagated into $(T^{-1} \cdot T_{Z \to Y}) \cdot Z \in (F_Q)^{C \times 1}$ with t' nonzero components. If $t' > t$, we say the error is propagated. We denote the error Z, which is the error occurs actually, with the terminology "original error". After the impact of the network coding, the error Z is propagated into $(T^{-1} \cdot T_{Z \to Y}) \cdot Z \in (F_Q)^{C \times 1}$. We call the error $(T^{-1} \cdot T_{Z \to Y}) \cdot Z$ "propagated error". Usually, the nonzero components of $(T^{-1} \cdot T_{Z \to Y}) \cdot Z$ are far more than t. This is the famous error propagation problem in the network coding. We can also interpret this problem from other perspective. This will help us understand the error propagation problem more clearly. We call the above model as the first model and the upcoming model as the second model. In this paper, we adopted the first model and the second model is just used to interpret the propagate problem more clearly. In the sense of transformation of generate matrix G, the transport procedure can be expressed by the simplified formulation $Y = T \cdot X = T \cdot G \cdot \kappa$. We can regard $T \cdot G$ as the generate matrix a new code Ω'. Because of the impact of T, the minimum distance $d_{\min}'$ of code Ω' is usually smaller than $d_{\min}$, i.e., $d_{\min}' \le d_{\min}$. The error $T_{Z \to Y} \cdot Z$ can be regarded as a disturbance to the new code Ω'. $t'' < d_{\min}'/2$ is the number of nonzero components in the vector $T_{Z \to Y} \cdot Z$. If $t'' < d_{\min}'/2$, we can decode κ successfully based on the decoding algorithm of code Ω'(not Ω). Therefore, we can conclude that the combination operation of network coding in the intermediate nodes makes it harder to decode κ. The issue in the network coding is more complicated than that in the point-to-point communication environment. In this environment, to recover the original uncompressible message κ, we should first perform network decoding and then perform decoding procedure of code Ω. The decoding difficulty not only comes from the combination of network coding, but also is leaded by the complexity of code Ω. The network coding scheme in the network and code Ω in the source must maintain good coordination and cooperation. The two codes should be constructed delicately.

To describe the model more conveniently, two terminologies are defined here. They are *original error Z* and *propagated error* $(T^{-1} \cdot T_{Z \to Y}) \cdot Z$. *Original error Z* is caused by many reasons. It includes the random error from physical cause and error caused by attacks from malicious nodes. It also includes the error caused by the rank deficiency of the transfer matrix of the source messages. The terminology of "original error" captures the essence that the error is injected to one link from the outside word. It is the equivalent of "symbol error" defined in [2]. It is also the equivalent of "corrupted packets" defined in [3] and "erroneous packets" in [4]. Another terminology is *propagated error* $((T^{-1} \cdot T_{Z \to Y}) \cdot Z)$ which represents the propagated result of the link error in the network coding. Under the influence of the error transfer matrix, the link error is enlarged to the propagated error. The definition of the two terminologies "original error" and "propagated error" are just right necessary.

Based on the brief description about error propagation in the network, it is easy to see that the key factor is to reduce the number of propagated error. That is to say we should make the value t' as small as possible. In $(T^{-1} \cdot T_{Z \to Y}) \cdot Z$, Z is objective and it cannot be modified. If we want to decrease t', $T^{-1} \cdot T_{Z \to Y}$ is the only target that we can change. $T^{-1} \cdot T_{Z \to Y}$ reflects the construction effect of network coding scheme. We have to construct network coding scheme delicately to avoid the spread of error Z. Just for a sink, it may appear $t' < t$, but for a multicast network, the minimum of t' among all the sinks is indeed bigger than t. If else, we can always reach the upper bound of multicast capacity, no matter how many original errors. This contradicts with common sense. We cannot construct such unrealistic perfect network coding scheme. Yang et al. [5] also confirm this point. Therefore, in the network coding, the best case is $t' = t$. We need to construct the network coding scheme delicately to reach this best case. Even so, this best case just appears in the coherent network if Hamming distance is adopted. The reason is that, in coherent network, we can take advantage of the topology which is a prior knowledge. That means we can construct $T^{-1} \cdot T_{Z \to Y}$ ingeniously to restrain the spread of Z. A coherent network is a network whose topology is known and stable. Accordingly, non-coherent network is a network whose topology is unknown. However, in non-coherent network, $T^{-1} \cdot T_{Z \to Y}$ is completely random. We cannot interfere with the construction of $T^{-1} \cdot T_{Z \to Y}$. The simulation experiment in the following section shows, even if $t = 1$, $T^{-1} \cdot T_{Z \to Y} \cdot t$ is always between $C/2$ and C when the max flow min cut is smaller than 7. If the max flow min cut is bigger than 7, even if $t = 1$, $T^{-1} \cdot T_{Z \to Y} \cdot t$ is always equal to C. In the latter situation, the propagated errors $(T^{-1} \cdot T_{Z \to Y}) \cdot Z$ always pollute all the components of received messages Y. It is not surprising to see such a phenomenon because $T^{-1} \cdot T_{Z \to Y}$ is completely random. Although some works increase the information rate as far as possible, the effectiveness of such improvements is limited [6, 7]. $T^{-1} \cdot T_{Z \to Y}$ has the potential to pollute the received messages completely. Therefore, the propagated error in network coding is dense. Especially, we have to face the fact that a propagated error mostly all the time pollute just 100% of the received packets. If we want to apply network coding scheme in a real application, an efficient NEC has to be proposed. Unfortunately, there is no such a method to solve this problem.

Though there are fruitful works for the error-correcting of network coding, most of them have a fatal drawback that they cannot correct corrupted propagated errors which are dense. As far as we know, all the existing works about NEC have an unrealistic assumption: the number of errors in network coding is bounded by a constant which is less than C. More so, in the situation where the traditional block codes are used, they assume that the number of errors in network coding is smaller than $C/2$. They cannot correct errors beyond $d_{\min}/2 = (C - k)/2 < C/2$ where (C, k) is MDS linear block code with code length C and information length k. Only a few studies refer to the assumption that the number of errors in network coding is bigger than $C/2$. They are list coding in network coding [8, 9] homomorphic signatures [10]. However, homomorphic signatures based on cryptographic approaches can correct the propagated error which equals C. However, cryptographic approaches have great complexity, and it is completely impractical in NEC [10]. Although a novel idea using nonlinear network coding seems a promising method to correct more errors, it cannot solve the problem completely just now and this method needs further being studied [11]. Except homomorphic signatures [10], there is no approach that can cope with propagated error which reaches C.

Next, we will give a review of previous research works. All the mentioned works will be judged from such a perspective: how many errors can be corrected by these methods at most? Can they correct the propagated error which reaches C? For limitation of space, we will just outline the most relevant and typical works about NEC. Sanna and Izquierdo [12] has made a survey on NEC.

1.2 Existing error-correcting methods in network coding

Existing solutions can be categorized into cryptographic approaches and information theoretic approaches. Generally speaking, cryptographic approaches have high rate, but they have high complexity. Meanwhile, information theoretic approaches have low complexity, but they cannot cope with the dense corrupted errors beyond $C/2$.

Cryptographic approaches include the methods such as keys, signatures, null space, and authentication [13]. And also, it is really worthwhile to mention the homomorphic signatures [10]. In the homomorphic signatures, the intermediate nodes can combine and encode the incoming hash packets and forward them without knowing the content of native packets or private key of the source node. The hash packet is a parity part of the common packet. The parity part is the so-called homomorphic signature. The excellent advantage of this scheme is that it can correct all the errors occurring in every links in network coding. That is to say it has solved the extreme tough problem: the network coding is susceptible to errors. However, almost all the existing homomorphic signature schemes have high complexity and intolerable delay [14]. The reason is that every intermediate node needs to perform time-consuming computation based on cryptographic approaches. Therefore, the homomorphic signatures are utterly impractical.

The existing works about information theoretic approaches for NEC can be divided into two categories. In the first category, a secret channel is needed. On the contrary, the secret channel is not needed. The representative work of the first type is [15]. After both parity symbols and hash values are sent over secret channels, the original messages can be recovered through solving equations. Parity symbols and hash values are used to counteract the uncertainty of the transfer matrix in the network coding. However, both the field size and the packet length are needed to be sufficiently large. A secret channel is also needed. Therefore, it would not be a promising practical scheme. The second kind type is *redundancy NEC* which introduces redundancy to the space domain. This method is also the so-called FEC (forward error correction) method. It is also the main stream in NEC. These skills can decode the code when the number of erasures and errors is within the minimum distance of the code. These skills are developed from traditional codes' methods that add redundancy in the time domain.

For a coherent network, there are such typical relevant work as follows. Based on the concept of *transfer matrix in network coding* defined in [1], Zhang presents a ground-breaking method that gives a framework of error-correcting in coherent networks [16]. Some important concepts such as *error vector* and *error pattern* are introduced. The transfer of messages and errors in the network coding can be expressed by the following equation.

$$Y = (\kappa A + z)(I - F)^T B \tag{1}$$

Here, A, B, and F denote the adjacency matrixes of the source node, the sink node, and the whole graph. κ is the source message injected into the network, and Y is the message received by the sink. I is the unit matrix. The error is considered as an $1 \times |E|$error vector $z \in (F_Q)^{1 \times |E|}$. The error pattern of z is a $1 \times |E|$ error vector ρ_z with unitary components in the corresponding nonzero components of z. Note that $Z \in (F_Q)^{t \times 1}$ (in Section 1.1) and $z \in (F_Q)^{1 \times |E|}$ are all the original error in essence. Some components in$z \in (F_Q)^{1 \times |E|}$ can be equal to zero, and no components in $z \in (F_Q)^{1 \times |E|}$ can be equal to zero. They just can be defined with different dimensions to adapt different algebraic formulations. *rank*(ρ_z) is the number of the nonzero components of z. It also generates traditional codes' concepts of the minimum distance and MDS (maximum distance separable) to NEC. It shows that if the coding field is big enough, there exists a linear network MDS code with minimum distance $d_{\min} = C - k + 1$ where $k$is the dimension of the information transmitted in the source. The $d_{\min}$of the generated NEC has properties like traditional codes. It can decode error z with $\text{rank}(\rho_z) \le 1/2^*(d_{\min} - 1)$. It also can decode erasure z with $\text{rank}(\rho_z) \le d_{\min} - 1$. The erasure is like that mentioned above, the error links, i.e., error pattern is known by the sink. Obviously, the MDS codes in [16] cannot correct dense corrupted errors beyond $C/2$ because $d_{\min} = C - k + 1$.

For the construction of MDS codes of NEC in coherent networks, [16] does not give an efficient construction algorithm. It just outlines this existence of MDS code in coherent networks. Its brute force decoding algorithm just checks all error patterns ρ in a non-decreasing order of cardinality up to $\text{rank}(\rho) \le 1/2^*(d_{\min} - 1)$ and then solve equations. Therefore, [17–20] each gives the construction algorithm of a MDS codes in NEC. Yang et al. [17] is based on the model of [16], and it just considers the situation that $\text{rank}(\rho) \le d_{\min} - 1 = C - k$ where $\text{rank}(\rho)$ is the number of the nonzero of ρ. Xuan et al. and Matsumoto [18, 19] all modify the Jaggi-Sanders algorithm [21] to get an efficient construction algorithm. In finding the global coding kernel for the processing link, they select a vector from the candidate vectors. Through an exhaustive search, the candidate vectors are promised as legal by avoiding all the error patterns ρ where $\text{rank}(\rho) \le d_{\min} - 1 = C - k$. Bahramgiri and Lahouti [20] is very similar to [18]. Except [17–20], there is no other important construction schemes for coherent networks with Hamming metric as far as we know. Especially, what deserves to be mentioned is that these error-correcting codes for coherent networks is based on Hamming distance. It handles symbol errors, rather than dimensional errors [2]. However, obviously, all the three schemes also cannot correct dense corrupted errors beyond $C/2$ because $\text{rank}(\rho) \le C - k$.

For non-coherent networks, there are mainly two kinds of methods essentially. They are subspace codes based on the subspace distance and rank codes based on rank distance. Silva and Kschischang present a seminal idea of subspace codes. They consider, for the completely random network channel resulting from random coefficient operations in the intermediate nodes, the only stable thing is the vector space spanned by the source messages. The subspace distance captures the above essence. Theorem 2 in [4] shows it can correct up to an *error dimension* $\lfloor \frac{D(\Omega)-1}{4} \rfloor$ where $D(\Omega)$ is the minimum space distance about the space code Ω. Therefore, a subspace code Ω can correct $\lfloor \frac{D(\Omega)-1}{4} \rfloor$ corrupted packets (it is original corrupted packets) at most. It is for that, in the worst case, t corrupted packets (it is original corrupted packets) can make $2t$ *error dimensions*. Therefore, the space code cannot correct more than $\lfloor \frac{D(\Omega)-1}{4} \rfloor$ errors.

Another approach is the rank code [22]. It can also correct no more than $d_{\min}/2$ corrupted packets where $d_{\min}$ is the minimum rank distance of the rank codes. The maximum value of $d_{\min}$ is C.

After we are aware of the shortcomings about the subspace distance in Silva and Kschischang's method, [2] combines correcting symbol errors and dimension losses together. Symbol errors are based on the Hamming metric, and dimension losses are based on the subspace distance metrics. However, the effectiveness of this improvement is limited. Skachek et al. [2] points that most of NEC approaches have a limited error-correcting ability. It also shows directly, rather than evading the questions and avoiding them, that the errors in network coding are really very dense. Mahdavifar and Vardy [9] generates Silva and Kschischang code about list decoding. It shows that for any L, the list-L decoder can correct at most $L - \frac{L^2(L+1)}{2} R$ errors where R is the normalized rate of the code. However, the solutions are not unique and additional redundancy must be send again. Therefore, it is also inefficient. Based on [9, 23], this improves the list model and achieves a much bigger decoding radius $\frac{C}{k}$-1 where C is the max flow min cut and k is the size of information. Guruswami and Sudan [24] can correct $C \cdot (1 - \sqrt{1 - \frac{d_{\min}}{C}})$ errors where

Hamming distance is adopted instead of subspace distance.

In conclusion, most approaches can correct few errors. However, it is not realistic to assume that, t, the number of the original error is less than $C/2$ or $\lfloor \frac{D(\Omega)-1}{4} \rfloor$ (in [16]). The original error usually exceeds $C/2$. Under a fixed BER (bit error rate), the more the links, the more the corrupted packets. The situation will be worse where no link-layer error correction is performed; this kind of network includes sensor networks where the computational power is not sufficiently large. The random errors are more common in reality indeed. Based on the NEC developed from traditional codes, the error-correcting ability cannot go beyond $d_{\min}/2$ according to the inexorable law. Even listing decoding is influenced by this law. Though list decoding can correct errors whose number is beyond $d_{\min}/2$, the ability of correcting more errors is at the cost that the solution is not unique. It needs the additional information to make the solution unique. The essential reason is that the decoding is performed in the framework of traditional block codes.

Let us take a look at this problem from a different perspective formulation sketchily. The perspective is how the original error is transferred into the propagated errors based on different distance metrics. Martínez-Peñas [25] compares Hamming metric and the rank metric in network coding. X is transmitted with network coding scheme. The transport procedure can be expressed by the formulation,

$$Y = T \cdot X + T_{Z \to Y} \cdot \mathrm{Z} \tag{2}$$

The propagated error is $(T^{-1} \cdot T_{Z \to Y}) \cdot \mathrm{Z} \in (F_Q)^{C \times 1}$. If the metric is Hamming distance, where the Hamming weight of Z is t, then the Hamming weight of $(T^{-1} \cdot T_{Z \to Y}) \cdot \mathrm{Z}$ cannot be restricted within a small range less than C. Theoretically, it may equal the dimension of X and Y, where X and Y are both $C \times 1$ dimensional vectors. It means Z pollutes all the received messages and the messages are spread over the networks. When the max flow min cut is bigger than 7, the coding field must be bigger than 7. If the size of coding field is bigger than 7, the Hamming weights of $(T^{-1} \cdot T_{Z \to Y}) \cdot \mathrm{Z}$ is equal to C with a very high probability, even if there is only one nonzero component. It means that a single error may pollute all the received messages almost all the time, if the Hamming distance metric is adopted in network coding. The propagated error is much greater than earlier thought. For coherent networks, it provides the Hamming weights of $(T^{-1} \cdot T_{Z \to Y}) \cdot \mathrm{Z}$ is equal to Z's through exhaustive searching all the error pattern. It means, based on the construction of MDS codes in coherent networks, it prevents the corrupted error Z from spreading and puts the corrupted errors into a "cage" [17–20]. This needs a prerequisite that the characteristics of the networks must be known a priori. Hamming distance is not suited for non-coherent NEC.

For subspace distance metric, Theorem 1 shows that

$$d_s(\langle TX \rangle, \langle Y \rangle) \leq 2rankT_{Z \to Y}Z \leq 2rankZ \leq 2t \tag{3}$$

Here, rank$Z$ is the rank of Z, which means t corrupted errors can make $2t$ dimension change at most even though the network is not known at all according to the space metric. For the rank distance metric, it is obvious that t corrupted errors can make t rank changes in the received messages at most. It is for that the rank of $(T^{-1} \cdot T_{Z \to Y}) \cdot \mathrm{Z}$ is no more than Z's even though the $T^{-1} \cdot T_{Z \to Y}$ is not known at all. Therefore, from the perspective of "rank" metric, the error Z will not spread. If Hamming metric is adopted, the error will spread. However, the spread of original errors in non-coherent networks is avoided by adopting the subspace metric or the rank metric; the number of original errors can be corrected is very small. For coherent networks, the number of original errors can be corrected is also very small. Constructing a MDS codes based on the Hamming metric is a tough work too, because there are too many combinations of the error patterns to make an exhaustive search [17–20]. When the network becomes larger, this method will completely fail.

A possible idea for NEC is to find a method to correct corrupted errors as many as possible, rather than errors less than $d_{\min}/2$. Avoiding this point of "correct more errors" may be unrealistic. We need a new error-correcting framework which is different from the block codes theory.

1.3 Correcting propagated errors in network coding via L1 minimization

Recently, there is a new trend which is applying the machine learning theory to communications [26]. Combing the compressed sensing with the network coding is intensively researched by a lot of work. Among them, [27, 28] are representatives. However, there are some drawbacks. It takes advantage of the related information in different sources If these messages in different sources are not correlative, it is hard to apply the compressed sensing. The common assumption in this kind of studies is that the coding vectors of network coding satisfy the restricted isometry property (RIP) of the compressive sensing. This assumption has no serious mathematics foundation and not been met all the time in experiments [29]. Its purpose is mainly to decrease sampling frequency rather than to communicate.

There are also some works of applying the convex optimization to correct errors. Wright and Ma [30] is the first important in this kind of works. It proposes a decoding algorithm which is completely different from decoding methods of the block codes. However, based on the experimental data's analysis, the error-correcting ability is nearly equal to that of the traditional codes. Therefore, this method cannot be used to correct dense errors like the propagated error in network coding. We noticed an interesting work whose result is very surprising and attractive in [30]. This work is developed from the work [31]. It suggests that the accurate and efficient recovery of sparse signals is possible even 100% of the observations are nearly corrupted. The error fraction which can be corrected is at most 50% in traditional codes. In formulation $d_{\min}/n = ((n-k)/2)/n = (1-k/n)/2$ where n is length of one code and k is the information rate, we can see, even k/n is 0, the $d_{\min}/n$ is 1/2 at most. This work is stirring and attractive. Its character of correcting dense error is inherently suited for solving the tough task to correct dense “propagated error” in network coding. Ganesh et al. and Qiu and Vaswani [32, 33] are other works whose performance may be near the performance of [30]. Except [32, 33], as far as we know, none of the other works in parse recovery have studied the case of any other kinds of large noise. However, in our work in this paper, we just consider applying [30] to the network coding.

The information does not need have any a priori characteristics inherently. Some redundancy of zeros is added to produce a sparse signal. Then, the sparse signal is sent through a dense error channel. Finally, the received signal polluted by the dense error can be corrected accurately via L1 minimization. The networks adopting network coding can be seen as such a channel with dense errors. Therefore, it is natural to apply [30] to the network coding.

Though near 100% errors can be corrected, in experiments, [30] can correct successfully varying the fraction of errors from 0 to 0.95. But when the fraction of errors is high, the information rate is low. It also gets a high information rate when the corrupted fraction is 60%. Theoretically, [30] cannot correct accurate (not near) 100% fraction of errors. Correcting 100% fraction of errors contradict the information theory and the basic truth. However, as mentioned above, when the size of the coding field is bigger than 7, the corrupted fraction is nearly 100%. Therefore, we should give some kinds of a priori interfaces to the network coding to decrease its corrupted percentage.

In the sense of the sparseness of L1 minimization in [30], we can make the transfer matrix sparse in random network coding. This method makes the “propagated errors” less than the “original errors”, even the latter's fraction is 1 (each link in networks has an original error). After this operation, the error-correcting scheme of the network coding will be performed successfully, regardless of how many the corrupted errors in networks there are.

However, the above method is just effective for the environment where the max flow min cut is smaller than 7. For the communication in the realistic world, the network whose min cut is bigger than 7 is the common environment. Therefore, we have to give some improvements based on L1 minimization in [30].

1.4 The potential viable solutions

It is impossible to avoid the two facts: one is that the propagated error will just pollute 100% of the observations in random network coding and the other is the max flow min cut of the most networks is bigger than 7. Applying the network coding technique to the real world needs us to find a new way of dealing with the propagated errors in random network coding. Because the coherent network is a model with unrealistic assumptions about real processes, we are not going to take the question of error correction for coherent network into consideration for the time being. We will only consider the error correction issue for random network coding. This problem is pregnant and agent.

Based on the above description, the existing methods to solve the error-correcting problem in network coding are divided into six categories. The first is cryptographic approaches whose representative work is homomorphic signatures [10]. The second is being based on Hamming distance to construct a NEC (network error correction coding) which is MDS (maximum distance separable). The third is being based on the rank distance [3] or subspace distance [4] to construct a NEC which is also MDS. The fourth is the list decoding. The representative work is [9]. Mahdavifar and Vardy [9] generate Silva and Kschischang code to list decoding. Especially what deserves to be mentioned most is that there are also list decoding methods based on rank distance [8] and Hamming distance [34, 35]. Unlike [9] which designs list decoding with a purpose to solve the error-correcting problem in random network coding, [8, 23, 24, 34, 35] are not designed for solving the error-correcting problem in random network coding. However, because [8, 23, 24, 34, 35] can correct errors beyond $d_{\min}/2$, they may have the potential to solve the propagated errors in random network coding. The fifth is the approach based on the secret channel. The representative work is [15]. The sixth is John Wright's method. The method has been introduced briefly in Section 1.3.

All the above methods have not solved the propagated error problems in network coding. However, we also want to select some methods among them as the potential approaches to solve this tough problem. Even if they do not work at it ultimately, they can offer some insight and enlightenment for us at least. Because of the obvious lack of utility, we will neglect approaches from the first to the third. The first approach has a high complexity.

The second and the third approaches can correct errors not beyond $d_{\min}/2$. Though the fourth approach can correct errors more than $d_{\min}/2$, the normalized rate of errors which are corrected by list coding is also far less than 100%. It also seems that the fifth has the potential to solve the propagated problem in random network coding. The rate of the approach with shared channel in [15] is $C - t - k^2/j$. The source encodes k unimpressive packets into one batch with $C - t$ redundancy and then sends the batch into network. A packet contains a sequence of j symbols from the finite field F_Q. If n is big enough, k^2/j will be a asymptotically negligible term. Therefore, the rate of the approach with shared channel in [15] is $C - t$ asymptotically. If t is big enough, we say, $t = C - 1$, the error will pollute nearly all the observations. Even if the error pollute nearly all the observations, this scheme also can correct the error by adding t redundancy packets and send both parity symbols and hash values over secret channel. The size of parity symbols and hash values reach k^2/j which can be neglected. However, the redundancy (t redundancy packets) is too much which results in a low information rate ultimately. What is nice is that this scheme can correct errors beyond $d_{\min}/2$ though the redundancy cost is expensive. Based on the traditional block code, we cannot correct errors beyond $d_{\min}/2$. However, the fifth method also cannot correct propagated error which is $t = C$. However, it certainly draws some inspiration and reference to the error correction when $t = C$.

Obviously, among the six methods, there is no*t* an alone method which can correct the dense errors in random network coding. One *is* inspire*d* with the combination of some methods to solve the difficult problem. John Wright's method is a method which seems has the greatest potential to solve the tough problem of errors spread in random network coding if some variable improvement solutions are given to [30]. However, some improvements must be given.

1.5 Combination of L1 minimization and list decoding

Though [30] can correct dense error which is nearly 100% of the observations, they also cannot correct completely 100% corrupted observations. The conclusion is that, if Hamming distance is adopted, the ratio of propagated error will be 100% in random network coding and there is no method which can correct so dense errors. The inspiration is to adopt list-decoding of subspace codes for error correction because the ratio of propagated error will be not 100% in random network coding in the sense of subspace distance. To overcome the drawback that the solution of list decoding is not unique, we need an "inner code" to achieve a unique solution from the multiple solutions of list decoding. However, the traditional block code will not be the candidate of the "inner code" because it will not correct errors exceed $d_{\min}/2$ where the errors in the "inner code" will exceed $d_{\min}/2$ usually. We will adopt the L1 minimization method of John Wright as the "inner code" which can correct errors more than $d_{\min}/2$. The list decoding code can be such codes as in [24] which can correct $C\cdot\left(1-\sqrt{1-\frac{\mathrm{d_{min}}}{C}}\right)$. Though $C\cdot\left(1-\sqrt{1-\frac{\mathrm{d_{min}}}{C}}\right)$ can approach C when $\mathrm{d_{min}}$ approaches $C/2$ and C is big, the propagated errors are always equal to C; [24] also cannot correct errors 100% of the propagated errors. Thus, we should adopt such list codes which are based on subspace or rank distance. In the sense of subspace distance, the propagated error usually does not pollute all the received messages. Thus, the list code based on subspace can correct propagated errors in random network coding though the solution is not unique. To make the solution unique, we combine the list code based on the subspace code and LI optimization together, and LI optimization can make the solution unique after the list decoding of the subspace code is performed. In one of the solutions after list decoding, there are some differences between the solution and the real unique solution and the difference may exceed $C/2$ errors. If L1 minimization method of John Wright be as the "inner code", it can correct difference which may exceed $C/2$ errors. The whole decoding procedure is completed. This is a perfect combination of different methods.

The remainder of this paper is organized as follows. Section 2 presents a brief review on [30] and gives some basic definitions. In Section 3, we will formally give our scheme. Then, Section 4 performs the experiments. Finally, Section 5 presents our conclusions.

2 Error-correcting model in John Wright's model and list decoding of subspace distance

2.1 Error-correcting model in John Wright's model

The flowing definitions mainly refer to [30]. Consider the problem of recovering a sparse signal $x_0 \in R^n$ from highly corrupted observations $x \in R^m$:

$$x = Ax_0 + e_0 \tag{4}$$

where $e_0 \in R^m$ is a sparse vector of errors with an arbitrary magnitude. The model for $A \in R^{m\times n}$ captures the idea that the messages consist of small deviations about a mean; hence, the model for A likes a "bouquet". A are i.i.d. samples from a Gaussian distribution:

$$A = [a_1 \dots a_n] \in R^{m\times n},\ a_i \sim_{iid} \mathbb{N}\left(\mu, \frac{v^2}{m}\mathrm{I_m}\right),$$
$$\|\mu\|_2 = 1, \qquad \|\mu\|_\infty \le \mathrm{C}_\mu m^{-1/2}. \tag{5}$$

The two assumptions on the mean force it to remain incoherent with the standard basis as $m \to \infty$.

Assumption 1 (weak proportional growth). A sequence of signal-error problems exhibits weak proportional growth with parameters $\delta > 0$, $\rho \in (0, 1)$, $C_0 > 0$, $\eta_0 > 0$, denoted $\mathrm{WPG}_{\delta,\rho,C_0,\eta_0}$, if as $m \to \infty$,

$$\frac{n}{m} \to \delta, \frac{\|e_0\|_0}{m} \to \rho, \|x_0\|_0 \le \mathrm{C_0 m}^{1-\eta_0} \tag{6}$$

We say the cross-and-bouquet model is ℓ^1 – recoverable at (I, J, σ) if for all $x_0 \ge 0$with supporting I and e_0 with supporting J and signs σ,

$$\begin{aligned}(x_0, e_0) = \ & \arg\min \|x\|_1 + \|e\|_1 \\ & \text{subject to } Ax + e = Ax_0 + e_0\end{aligned} \tag{7}$$

And the minimization is uniquely defined.

Theorem 1 For any $\delta > 0$, $\exists\, v_0(\delta) > 0$ such that if $v < v_0$ and $\rho < 1$, in $\mathrm{WPG}_{\delta,\rho,C_0,\eta_0}$ with a distribution according to (4), if the error support J and signs σ are chosen uniformly at random, then as $m \to \infty$,

$$P_{A,J,\sigma}\left[\ell^1\text{–recoverable at } (I, J, \sigma) \forall I \in \binom{[n]}{k_1}\right] \to 1 \tag{8}$$

In other words, as long as the bouquet is sufficiently tight, asymptotically ℓ^1 minimization recovers any sparse signal from almost any errors with support size less than 100%.

2.2 List decoding of subspace distance

2.2.1 List decoding

In the traditional linear block code, it cannot correct errors which exceed $d_{\min}/2$. The so-called list decoding, proposed independently by Elias in the late 50s, allows the decoder to output a list of all code words that differ from the received word in a certain number of positions. Even when constrained to output a relatively small number of answers, list decoding permits recovery from errors well beyond the $d_{\min}/2$ barrier and opens up the possibility of meaningful error correction from large amounts of noise [36].

Definition 1 ((e, L)-list decodability) For positive integers e, L, a code $C \subseteq \mathrm{F}_q^n$ is said to be (e, L)-list decodable if every Hamming ball of radius e has at most L code words, i.e., $\forall x \in \mathrm{F}_q^n$, $|B_q(x, e) \cap C| \le L$.

2.2.2 Subspace distance

Koetter and Kschischang [4] proposes the subspace code which is dependent on the subspace distance other than the Hamming distance. Motivated by the property that linear network coding is vector space-preserving, information transmission is modeled as the injection into the network of a basis for a vector space V and the collection by the receiver of a basis for a vector space U. Based on the subspace distance, the original error in random network will not spread, and this is a good characteristic in the error correction of random network coding.

Definition 2 The subspace distance is

$$d(A, B) = \dim(A + B) - \dim(A \cap B) \tag{9}$$

it is also

$$d(A, B) = 2\dim(A + B) - \dim(A) - \dim(B) \tag{10}$$

Definition 3 A subspace code denoted as Ω has a minimum distance as $D(\Omega) = \min_{X,Y \in \Omega : X \ne Y} d(X, Y)$.

The maximum dimension of code words of Ω is denoted by $\ell(\Omega) = \max_{X \in \Omega} \dim(X)$.

2.2.3 The combination of list decoding and subspace distance

Combining list decoding and subspace code, [23] proposes list decoding of subspace codes. For any integers L and r, the list-L decoder with multiplicity$r$ guarantees successful recovery of the message subspace provided that the normalized dimension of error is at most $\frac{2(L+1)}{r+1} - 1 - \frac{L(L+1)}{r(r+1)} R^*$ where R^* is the normalized packet rate. In the network coding environment, $R^* = \frac{k}{C}$. The decoding radius of this scheme with an appropriate choice of r approaches $\frac{C}{k} - 1$. Obviously, $\frac{C}{k} - 1$ can be very large and can even approach to $C - 1$. The big decoding radius is very suitable for the propagated error correction in network coding.

3 Our proposal

3.1 Important issues needed to be clarified

A sketch approach is as follows: $\kappa \in {F_{q^p}}^{k \times 1}$ is the message needed to be sent where $k < n$. κ may not sparse. Add $n - k$ zeros to κ to form $x_0 \in {F_{q^p}}^{n \times 1}$ which is sparse. Then, get $x \in {F_{q^p}}^{m \times 1}$ based on Eq. (4). Although adopting complex field may improve performance of network coding [37], we just consider real field other than complex field. In this case, p is an integer and F_{q^p} is a finite field.

Then $x \in {F_{q^p}}^{m \times 1}$ is coded with a subspace code $\Omega \in {F_{q^p}}^{C \times m}$ which has a coding matrix $G = {F_{q^p}}^{C \times m}$. $x' = G \cdot x$ is generated. This coding procedure is performed in the field F_{q^p}.

Next, x' is been sent through the network with network coding scheme. That is to say, x' is sent through the networks with network coding and polluted by errors of the networks. The received messages are y which is the mixture of x' and errors. This process can be expressed by Eq. (2) where X is replaced by x'. That is,

$$y = T \cdot x' + T_{Z\to y} \cdot Z \tag{11}$$

L1 optimization is performed in F_{q^p} field. Therefore, all the encodings in intermediate nodes are performed in F_{q^p}. The decoding of L1 optimization is also performed in F_{q^p}. However, all the coefficients of network coding are selected in a finite field F_q as a usual way of doing the common network coding.

With some abuse of terminologies, we re-define the dimensions in (10), and they are different from (2). $T \in C \times C$. T is the true transfer matrix of x'. $Z \in R^{t\times 1}$ is the error vector. Note that, in our L1 optimization method, there is no need to assume $t \le C/2$ as done in the previous works. t is a big arbitrary and even equals the number of all the links. This is contradictory to intuition seemingly. The propagated errors are what the t original errors are projected to the received messages Y.

Z's all components are nonzero. t is the number of corrupted packets. $T_{Z\to y}$ refers to the linear transform from error edges to the sink. $T$is $C \times C$, and $T_{Z\to y}$ is $C \times$ t.

3.2 The transfer model in non-coherent network

In Section 1, we have described the transfer model in the coherent network. The transfer model in the non-coherent network is different from the transfer model in the coherent network. We should clarify this model in detail because it is important for the network coding and decoding. A classical random network code indicates that y includes the identity matrix as a part of each batch. The identity matrix sent by source experiences the same transform matrix T with the raw data of the batch. Thus,

$$\hat{T} = T \cdot I + T_{Z\to y} \cdot L \tag{12}$$

where $\hat{T}$ and L are the columns corresponding to I's location in Y and Z respectively. $\hat{T}$ is $C \times C$, and L is t $\times C$. By substituting T, (10) can be simplified as:

$$y = \hat{T} \cdot x' + T_{Z\to y} \cdot (Z - L \cdot x') \tag{13}$$

Note that the matrix $\hat{T}$ acts as a proxy transfer matrix for T, which the sink does not know. Note that the above is mainly in reference to [15]. Equation (13) is slightly different from $y = T \cdot x' + T_{Z\to y} \cdot Z$ which is for coherent network. Equation (13) is for random network coding. In random network, $T$is unknown and it is replaced by $\hat{T}$. $y = T \cdot x' + T_{Z\to y} \cdot Z$ is degraded from Eq. (13) for random networks. In coherent networks, there are no errors in the header because there are no coding vectors in the head of the packets. Therefore, L is a 0 matrix.

In the sink, packets are collected until the proxy transfer matrix $\hat{T}$ is invertible. Matrix $\hat{T}^{-1}$ is left multiplied in Eq. (13); we get

$$x' = \hat{T}^{-1} \cdot y - \hat{T}^{-1} \cdot T_{Z\to y} \cdot \left(Z - L \cdot x'\right) \tag{14}$$

where $\hat{T}^{-1} \cdot Y$ can be obtained and $\hat{T}^{-1} \cdot T_{Z\to y} \cdot (Z - L \cdot x')$ is unknown. Let $(x')^d = \hat{T}^{-1} \cdot y$. $(x')^d$ is the result of network coding decoding in the sink. $(x')^d$ can be regarded as a deviation value of x'. In principle, $\hat{T}$ can be seen as a proxy transfer matrix of the true transfer matrix T to perform decoding.

However, there is a difference of $\hat{T}^{-1} \cdot T_{Z\to y} \cdot (Z - L \cdot x')$ between $(x')^d$ and x'. The difference is needed to be corrected through L1 optimization in [30], rather than traditional codes. Above all, the number of "original error" is Z, and the number of "propagated error" is $\hat{T}^{-1} \cdot T_{Z\to y} \cdot (Z - L \cdot x')$. The errors in the header mentioned above is expressed by L. $\hat{T}^{-1} \cdot T_{Z\to y} \cdot (Z - L \cdot x')$, which is $C \times 1$, represents the spread result of Z. In random network coding, we just know $\hat{T}^{-1}$. However, $T_{Z\to y}$, Z, L, and x' are all unknown. Theoretically, even t, the number of original corrupted packets, is very small, $\hat{T}^{-1} \cdot T_{Z\to Y} \cdot (Z - L \cdot x')$ is also potential to have C nonzero components. That is to say, $\hat{T}^{-1} \cdot T_{Z\to y} \cdot (Z - L \cdot x')$ pollutes every symbol of the messages $(x')^d$. With $wr(\beta)$ denoting the number of nonzero components (or symbols) in an arbitrary vector or matrix β and with $wr_{\text{norm}}(\beta) \in [0, 1]$ denoting the normalized $wr(\beta)$, if $wr_{\text{norm}}(\hat{T}^{-1} \cdot T_{Z\to y} \cdot (Z - L \cdot x'))$ is 1, where the percentage of propagated errors is 100%, we cannot decode successfully with [30]. If $wr_{\text{norm}}(\hat{T}^{-1} \cdot T_{Z\to y} \cdot (Z - L \cdot x'))$ is high, for example, 0.99999, [30] can decode successfully with a large m. However, the information rate is very low. It accords with the truth: the more the errors, the lower the information rate. Any method cannot contradict this basic truth. Therefore, in random networks, we just can control the sparseness of $\hat{T}^{-1}$ partly. In a finite field F_{q^p}, perform encoding procedure of network coding scheme in every intermediate node. In the sink, the received message is $y = T \cdot x' + T_{Z\to y} \cdot Z$. In the sink, perform the decoding algorithm of network coding $(x')^d = \hat{T}^{-1} \cdot y$. Perform the list decoding of subspace code Ω and get a list of solutions x^d which may have more than $C/2$ positions different from the real solution x. Perform L1 optimization of John Wright's scheme based on x^d and get x_0. Select the first $k$symbols of x_0 as κ. The so-called L1 Optimization and List Decoding of

Subspace Codes (L1OLDSC) Algorithm is illustrated in Algorithm 1.

Algorithm 1 (L1OLDSC) Algorithm

```
Step 1 Set involved parameters in L1 optimization according the Eqs. (5) and (6).
Step 2 In real field $F_{q^p}$, add $n-k$ zeros to $\kappa$ behind of it, to form $x_0 \in R^{n\times 1}$. Get $y \in R^{(m+n)\times 1}$ based on Equation $x = A \times x_0$.
Step 3 $x' = G \cdot x$ where $G = {F_{q^p}}^{C\times m}$ is the generating matrix of a subspace code $\Omega$.
Step 4 In a finite field $F_{q^p}$, perform encoding procedure of network coding scheme in every intermediate node. In the sink, the received message is $y = T \cdot x' + T_{Z\rightarrow y} \cdot Z$.
Step 5 In the sink, perform the decoding algorithm of network coding $(x')^d = \hat{T}^{-1} \cdot y$.
Step 6 Perform the list decoding of subspace code $\Omega$ and get a list of solutions $x^d$ which may have more than $C/2$ positions different from the real solution$x$.
Step7 Perform L1 optimization of John Wright's scheme based on $x^d$ and get $x_0$.
Step 8 Select the first $k$ symbols of $x_0$ as $\kappa$.
```

3.3 The notes on Algorithm 1

There are some notes on Algorithm 1. As long as the error rate is deceased less than 100% (not equal 100%), we can apply the L1 minimization methods in [30] to perform error correction.

The first note is that the overall information rate is very low. The original information is wrapped twice. The inner code is the L1 optimization, and the outer code is the subspace code. However, because the propagated error in network coding is very difficult, there are no other valid methods to solve this problem. Our method is of theoretical value.

The second note is why we choose to do the decoding of network coding scheme. A potential choose is to omit this step and we can perform the list decoding of subspace on y other than $(x')^d = \hat{T}^{-1} \cdot y$. In the sense of Hamming distance, y and $(x')^d = \hat{T}^{-1} \cdot y$ are almost 100% polluted. In the sense of subspace distance, $(x')^d = \hat{T}^{-1} \cdot y$ will not be 100% polluted and this is an advantage for the error correction in random network coding. In $x' = \hat{T}^{-1} \cdot y - \hat{T}^{-1} \cdot T_{Z\rightarrow y} \cdot (Z - L \cdot x')$, the propagated error is $\hat{T}^{-1} \cdot T_{Z\rightarrow y} \cdot (Z - L \cdot x')$ which will not pollute 100% of $(x')^d = \hat{T}^{-1} \cdot y$. However, if we adopt y other than $(x')^d = \hat{T}^{-1} \cdot y$, the errors will pollute 100% of the received packets even if the subspace distance is adopted. $T \cdot x'$ is a multiple to x', and this will introduce more "pollutions" than what is introduced to $(x')^d$ by $\hat{T}^{-1} \cdot T_{Z\rightarrow y} \cdot (Z - L \cdot x')$ in the sense of subspace distance. $T \cdot x'$ is a multiple style pollution, and $\hat{T}^{-1} \cdot T_{Z\rightarrow y} \cdot (Z - L \cdot x')$ is the pollution coming from the addition operation. Therefore, we should not omit the procedure about decoding of network coding.

The third note is what the parameter combinations of m and n are. The good parameter combinations are $m = 800$ and $n = 200$. However, if m is big, we have to offer a big C which is bigger than m. In real environment, there is usually no such big C. Therefore, it is better to choose a small m, but small m will lead to the decline of the effectiveness about John Wright's scheme. We can take a compromise between the effectiveness of John Wright's scheme and a small C. This point will not decline the theoretical significance of our scheme.

The fourth note is that is it better to reorganize $y \in R^{m\times 1}$ to $y' \in R^{C\times \frac{m}{C}}$, and then send these divided groups through network with network coding many times. The size of packets in one group C can be smaller than m. That is to say, the max flow min cut will not be limited by m and can be small. Therefore, our model can have generality. However, in this case, how many errors introduced to the received packets is unknown because reorganization will introduce new errors. If the new errors introduced by the reorganization will not pollute all the received packets in the sense of subspace distance, the reorganization will be worthy, or not, it is more harm than good. In this document, we will not consider the reorganization and we will just focus on the theoretical significance of this model and omitted the details.

The fifth note is what is the metric of L1 minimization in John Wright's scheme. The metric is Hamming metric other than subspace distance adopted in the "outer code" which performs list decoding of subspace codes. After the list decoding of subspace codes, there are some finite subspace or rank difference between the list solution and the real solution. The number of difference rank between the list solution and the real solution are in direct proportion to L. There is an empirical rule that a low-rank matrix is usually sparse. Taking advantage of this property, we can perform L1 minimization to recover the original message after the list decoding of subspace codes which commits the low-rank characteristics. If there is no step of list decoding of subspace codes, the matrix which corresponds to the message will not be of low rank in the sense of subspace distance and therefore will not be sparse in Hamming metric. Therefore, the two decoding procedures about list decoding based on subspace distance and L1 minimization based on Hamming distance are all needed.

4 Numerical results and discussions

In this section, numerical results are presented to further examine and verify the analytical results mentioned above.

The results include issues which are how the original errors spread, the effect of L1 optimization, and optimal parameters to increase the information rate. We make an additional discussion to end this section.

4.1 The original error propagation in the sense of hamming distance

For non-coherent networks, in $\hat{T}^{-1}\cdot T_{Z\to y}\cdot(Z{-}L\cdot x')$, we only can select $\hat{T}^{-1}$ other than $T_{Z\to Y}\cdot(Z-Ly)$ in the sink. In the sink, we construct the matrix T after receiving C coding vector. Thus, T is an exogenous variable. Naturally, $\hat{T}^{-1}$ is also an exogenous variable. With the size of coding field becoming bigger, the number of nonzero components in $\hat{T}^{-1}\cdot T_{Z\to y}\cdot(Z{-}L\cdot x')$ will be fast equal to C. That is to say, the propagate errors pollute all the received messages in the sink. We will investigate how the original error spread in non-coherent network through the experiment. Assume $wr(\hat{T}^{-1}\cdot T_{Z\to y}\cdot(Z{-}L\cdot x'))=C$, which is also the worst situation. With the above means, the errors will be propagated to the whole network.

Figure 1 shows, if randomly selecting coefficients of the local coding kernel, the received messages are nearly all polluted. But when the size of network coding field is smaller than 7, some received messages are not polluted. Theoretically, the bigger the size of the coding finite field, the more the chance that a symbol in this field is nonzero. Therefore, $\hat{T}^{-1}$ will be very dense if the size of coding field is big.

Through the above experiments, we can see the situation about the error spread in the network coding is serious. Especially, in random network coding, the propagated error $\hat{T}^{-1}\cdot T_{Z\to Y}\cdot(Z{-}L\cdot x')$ always pollutes all the received messages when the coding field is bigger than 7. Thus, we have to face such pessimistic fact and propose an effective method to confront such situation.

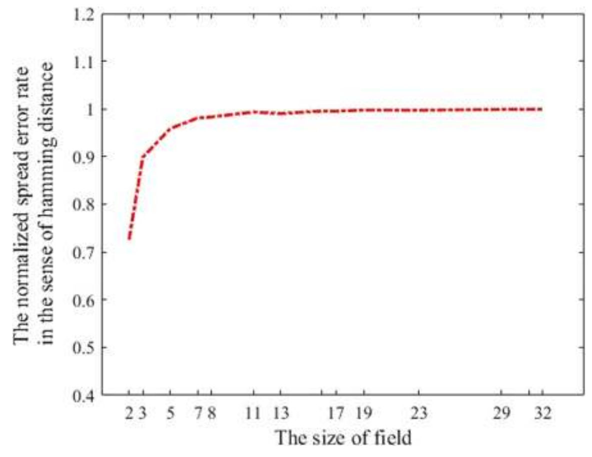

Fig. 1 The spread of errors in the sense of Hamming distance

4.2 The original error propagation in the sense of subspace distance

For non-coherent networks, if the subspace distance is adopted, the propagated error will not be polluted 100% of the received packets which is better than Hamming distance. In Fig. 2 we investigate the propagated way of errors in the sense of subspace distance. In the above section, we have analyzed the propagation of errors in the sense of subspace distance. Figure 2a shows the case with a smaller C while Fig. 2b shows the case with a bigger C. It shows that, compared with Hamming metric, subspace distance has more advantage. In the sense of subspace distance, the error propagation is alleviated considerably.

4.3 The effect of L1 optimization in network random coding

We will investigate the performance of the Algorithm 1. $\|x_0\|$ is on behalf of the sparseness of vector x_0. In

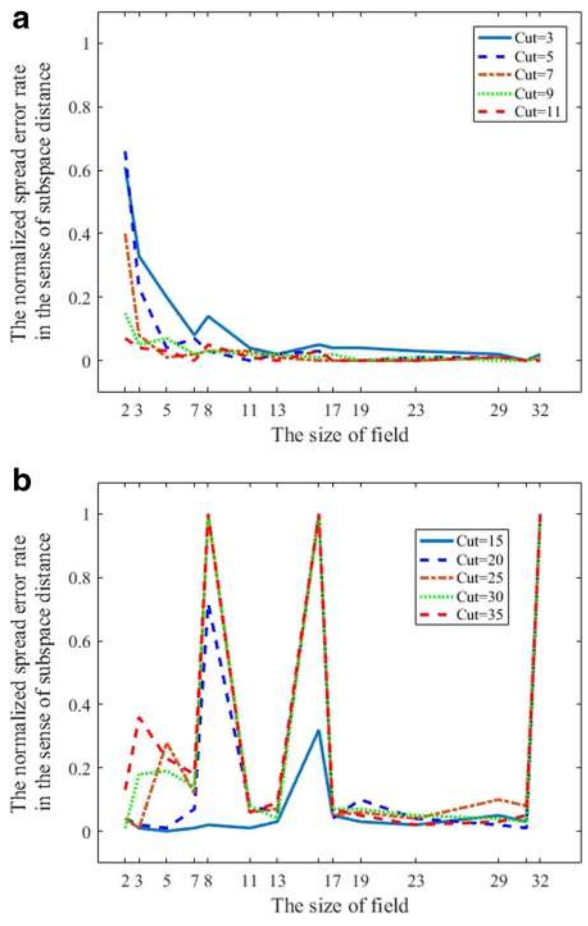

Fig. 2 The spread of errors in the sense of subspace distance. Legend ('M = 100', 'M = 200', 'M = 400', 'M = 800')

Algorithm 1, if the components of the original uncompressible messages κ have no zeroes, $\|x_0\| = k/n$.

Most of the parameters are the same with the simulation in [30]. The parameters are as follows: $\upsilon = 0.05$, $\delta = 0.25$, and $m \in \{100, 200, 400, 800\}$. We have not got any channel to get the original implement details of [30]. Because some implement details may be different, our effectiveness is just a little worse. However, the whole trend is the same.

In Fig. 3, $\|x_0\| = 1$ and we can see that the percentage of successful recovery and the fraction of corrupted errors are in inverse proportion. When m increases, the correcting fraction τ also increase and almost approaches 1. This point is surprising and attractive. It can be naturally adopted to correct the dense propagated errors in network coding, even 0.95 density errors can be recovered. However, the successful correction fraction is not satisfactory when error density is high, for example, 0.95. But we can increase the m to increase the successful correction ratio. Generally, when $m = 800$ and the fraction of errors is 0.6, the fraction of successful correction approaches 1. This certainly can meet the need in real communication. However, $\|x_0\| = 1$ also means a low information rate here. However, when the fraction of corrupted errors is 100%, this algorithm cannot recover the original uncompressible messages. Especially, when the fraction of propagated errors is 100%, we can also correct it sometimes.

If $\|x_0\| = 1$, the information rate will very be low. We will also investigate the performance of Algorithm 1 at different $\|x_0\|$. In Fig. 4, $\|x_0\| = m^{1/2}$. The high fraction of successful decoding is at the cost of the low information rate. If we want to increase the information rate, the fraction of successful decoding will be down. However, even when the information rate is higher, L1 optimization also has a surprising high fraction of corrected errors. In traditional codes, the fraction of corrected errors is 0.5 at most when the information rate approaches 0. The fraction of successful decoding is approximately 0.47. It is also higher than traditional codes, i.e., to a considerable information rate.

Both $\|x_0\| = 1$ and $\|x_0\| = m^{1/2}$ are extreme situations. It would be better to keep a balance between error-correcting ratio and information rate. It shows, when $\|x_0\|$ increases, i.e., higher information rate, the fraction of correction becomes lower. However, the fraction of correction is also acceptable.

4.4 Set other parameters to increase the information rate in L1 optimization

In [30], a better parameter about m is as $m \in \{100, 200, 400, 800\}$. As we say in the above, a bigger m will lead to a bigger C which is unrealistic. Thus, we want to investigate the L1 optimization with a smaller m. Figure 5 is the situation where $m \in \{8, 20, 40, 60, 80\}$ and $\|x_0\| = 1$. Figure 6 is the situation where $m \in \{8, 20, 40, 60, 80\}$ and $\|x_0\| = m^{1/2}$. We can see that even if a smaller m also will achieve a considerable good effectiveness. Therefore, it is feasible to make our scheme more generable by using a smaller m.

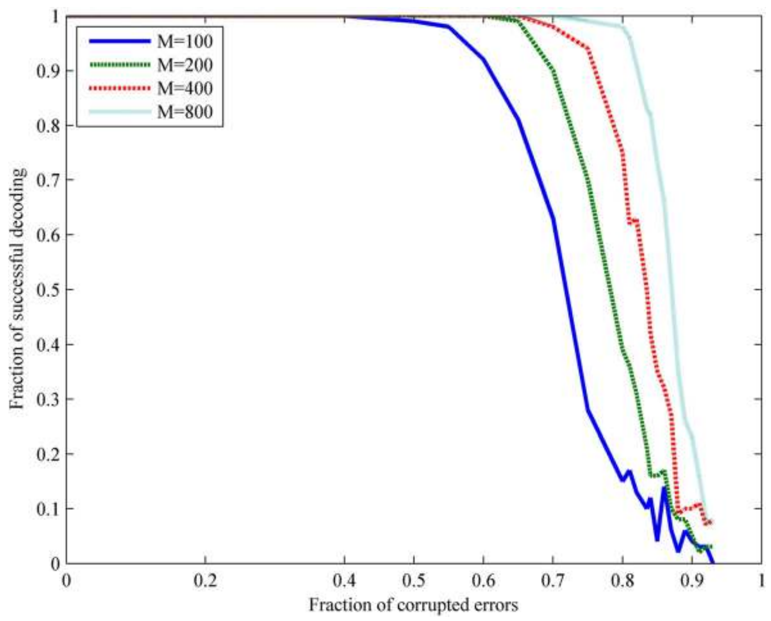

Fig. 3 Error correction in L1 optimization when $\|x_0\| = 1$. Legend ('M = 100', 'M = 200', 'M = 400', 'M = 800')

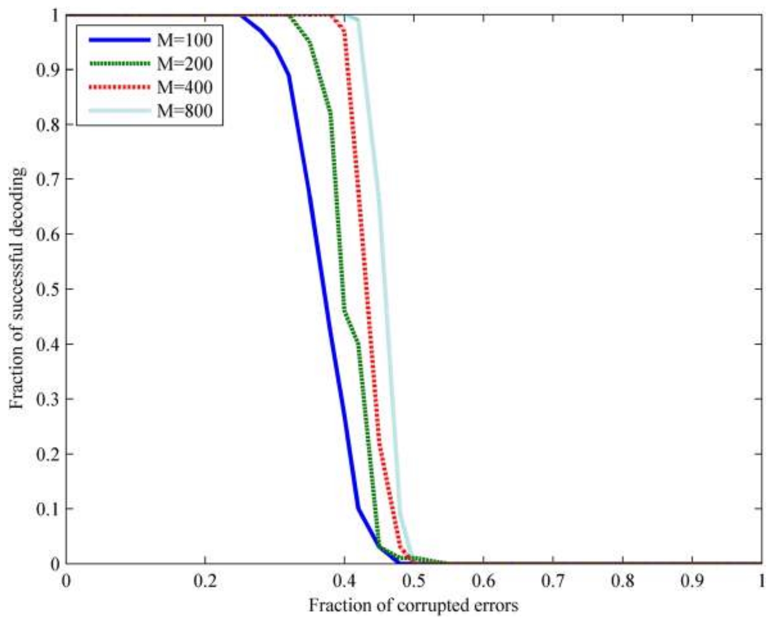

Fig. 4 Error correction in L1 optimization when $\|x_0\| = m^{1/2}$. Legend ('M = 8', 'M = 20', 'M = 40', 'M = 60', 'M = 80')

4.5 Discussions

For the numerical results about how the original errors spread, they consist with the facts. When the size of the coding field is big, the randomness will increase. Thus, all the observations are polluted completely. For the effect of L1 optimization to increase the fraction of the error correction, the more the signal is sparse, the higher the fractions of the error correction are. Even though nearly all the observations are polluted, numerical results show our algorithm also can recover the original messages. For optimal parameters to increase the information rate, numerical results show $m = 800$ and $n = 200$ are the optimal parameters to correct errors. However, with these parameters, the rate is low. We can achieve a higher information rate in some little penalty of the correction fraction. If the parameters are set suitably, it is possible to achieve an optimal combination of information rate and error correction fraction.

We can see that in the sense of subspace distance, the propagated errors will not pollute 100% of received packets which is different with the way of Hamming distance. Therefore, list decoding of subspace has the chance to perform decoding procedure successfully. Though the solution is not unique, L1 optimization

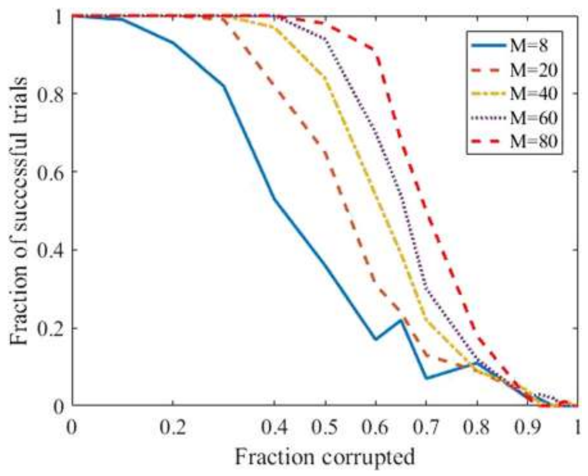

Fig. 5 Error correction in L1 optimization when $m \in \{8, 20, 40, 60, 80\}$ and $\|x_0\| = m^{1/2}$. Legend ('M = 8', 'M = 20', 'M = 40', 'M = 60', 'M = 80')

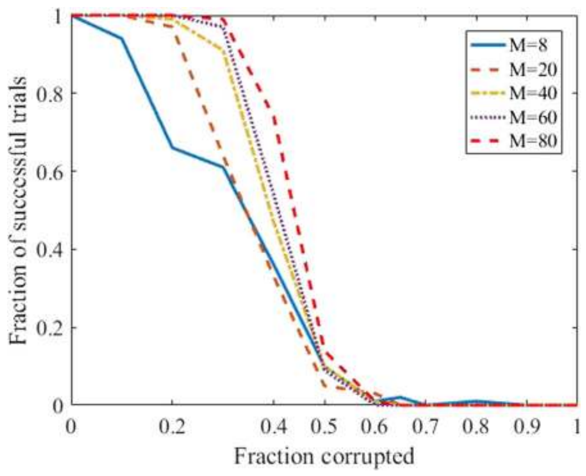

Fig. 6 Error correction in L1 optimization when $m \in \{8, 20, 40, 60, 80\}$ and $\|x_0\| = 1$

can decode and get the unique solution successfully. Figures 3 and 4 have proven this point.

We can also choose a small m and will not decline the theoretical significance of our scheme. Figures 5 and 6 prove this point.

Overall, the numerical results and observations in this section are consistent with our expectations.

5 Conclusions

We propose a new framework about the network error correction coding in random network coding. The scheme introduces the combination of L1 optimization method and list decoding to correct the propagated dense error in the random network coding. This method overcomes the shortcoming that the traditional block codes can correct corrupted errors no more than $C/2$ in random network coding. The experiments show our scheme can solve the error spread problem in random network coding. The rank code or the subspace code are the best cure methods to solve the error spread in the random network coding at present. Whether the rank code or the subspace code, they just can solve the error spread problem in the random network where the number of original error is less than $C/2$. Our scheme not only can correct the errors more than $C/2$ but also can correct errors which just pollute all the links. Although the information rate is low when the fraction of errors is near 100%, the information rate is acceptable if there is a small penalty in the fraction of successful decoding. Our scheme is also efficient in time. The decoding time of L1 optimization algorithm is no more than that of the decoding algorithm about the traditional block codes. Our scheme outperforms the existing network error correction coding schemes in random network coding. The scheme, which is the combination of L1 optimization and list decoding, has great significance to applying the network coding theory to engineering practice.

Acknowledgements
We thank the reviewer for helping us to improve this paper. We also thank the post-doctor Ximeng Liu of Nanyang Technological University for his valuable advice about paper writing. The bachelor candidate Wang Yu assisted with the laboratory experiment work, and we thank her for the support.

Funding
This work is supported by the fundamental research funds for Heilongjiang provincial universities (the study on error spreading depression in network coding, 2017-KYYWF-0719) which afforded the publication fee for the papers, Suihua technology office program (SHKJ2015-015, SHKJ2015-014) which also afforded the publication fee for the papers, National Science foundation of China (61571150,61603420) which afforded the fees for participating academic conference, and Education Office of Heilongjiang Province science and technology program(2016-KYYWF-0937) which also afforded fees for participating the academic conference Suihua College 2017 university-level key built discipline: computer science and technology program.

Authors' contributions
GZ and SC conceived the original idea almost at the same time when they are discussing the problem. GZ designed the algorithm of the experiment. SC organized this work and wrote this paper. DZ provided the research grant for this work and also assisted the first author in the minor revision. WS and CM replaced the secret channel method with list decoding of subspace codes. The new added method is proposed by WS and CM, and MN and XD assisted with the laboratory work and analyzed the results. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Author details
[1]Computer Science Department, Harbin Engineering University, Harbin 150001, China. [2]Suihua University, Suihua 152000, China. [3]Computer Science Department, Huaqiao University, Xiamen 361021, China. [4]School of Education Science, Nanjing Normal University, Nanjing 210023, China. [5]College of Computer Science, South-Central University for Nationalities, Wuhan 430074, China.

Received: 7 November 2017 Accepted: 11 March 2018
Published online: 27 March 2018

References
1. R Koetter, M Medard, An algebraic approach to network coding. *IEEE International Symposium on Information Theory, 2001* Proceedings. **11**(5), 104 (2002)
2. V Skachek, O Milenkovic, A Nedić, Hybrid noncoherent network coding. IEEE Trans. Inf. Theory **59**, 3317–3331 (2013)
3. D Silva, FR Kschischang, R Koetter, A rank-metric approach to error control in random network coding. Inf. Theory IEEE Trans. **54**, 3951–3967 (2008)
4. R Koetter, FR Kschischang, Coding for errors and erasures in random network coding. IEEE Trans. Inf. Theory **54**, 3579–3591 (2007)
5. S Yang, RW Yeung, Z Zhang, Weight properties of network codes. Eur. Trans. Telecommun. **19**, 371–383 (2012)
6. G Zhang, S Cai, Secure error-correcting (SEC) schemes for network coding through McEliece cryptosystem. Clust. Comput., 1–9 (2017). https://doi.org/10.1007/s10586-017-1294-5
7. G Zhang, S Cai, Universal secure error-correcting (SEC) schemes for network coding via McEliece cryptosystem based on QC-LDPC codes. Clust. Comput., 1–12 (2017). https://doi.org/10.1007/s10586-017-1354-x
8. V Guruswami, C Wang, C Xing. Explicit List-Decodable Rank-Metric and Subspace Codes via Subspace Designs. IEEE Transac. Info. Theory. **62**(5), 2707–2718 (2016).
9. H Mahdavifar, A Vardy, Algebraic list-decoding on the operator channel. IEEE Int. Symp. Inf. Theory Proc. **41**(3), 1193–1197 (2010)
10. D Charles, K Jain, K Lauter, Signatures for network coding. Inf. Sci. Syst. **2006**(1), 857–863 (2006)
11. G Zhang, S Cai, D Zhang, The nonlinear network coding and its application in error-correcting codes. Wirel. Pers. Commun., 1–30 (2017) https://doi.org/10.1007/s11277-017-5109-z
12. M Sanna, E Izquierdo, A survey of linear network coding and network error correction code constructions and algorithms. Int. J. Digital Multimedia Broadcasting. **2011**,1687–7578 (2011).
13. E Kehdi, B Li, Null keys: limiting malicious attacks via null space properties of network coding. Proceedings IEEE INFOCOM. 1224–1232 (2009)
14. Z Yu, Y Wei, B Ramkumar, Y Guan, An efficient signature-based scheme for securing network coding against pollution attacks. Infocom. Conf. Comput. Commun. IEEE, 1409–1417 (2008)
15. S Jaggi, M Langberg, S Katti, T Ho, D Katabi, M Médard, et al., Resilient network coding in the presence of Byzantine adversaries. INFOCOM 2007. 26th IEEE Int. Conf. Comput. Commun. **54**(6), 616–624 (2007)
16. Z Zhang, Linear network error correction codes in packet networks. IEEE Trans. Inf. Theory **54**, 209–218 (2008)
17. S Yang, RW Yeung, KN Chi, Refined coding bounds and code constructions for coherent network error correction. IEEE Trans. Inf. Theory **57**, 1409–1424 (2010)
18. G Xuan, FW Fu, Z Zhang, Construction of network error correction codes in packet networks. Int. Symp. Netw. Coding. **59**(2), 1–6 (2011)

19. R Matsumoto, *Construction Algorithm for Network Error-Correcting Codes Attaining the Singleton Bound* (Oxford University Press, Oxford, 2007), pp. 1729–1735
20. H Bahramgiri, F Lahouti, Robust network coding against path failures. IET Commun. **4**, 272–284 (2010)
21. P Sanders, S Egner, L Tolhuizen, Polynomial time algorithms for network information flow. Fifteenth ACM Symp. Parallel Algorithms Arch, 286–294 (2003)
22. D Silva, FR Kschischang, Using rank-metric codes for error correction in random network coding. IEEE Int. Symp. Inf. Theory, 796–800 (2007)
23. H Mahdavifar, A Vardy, Algebraic list-decoding of subspace codes with multiplicities. Commun. Control. Comput. **59**(12), 1430–1437 (2011)
24. Guruswami V, Sudan M. Improved decoding of Reed-Solomon and algebraic-geometry codes[J]. IEEE Transactions on Information Theory. **45**(6): 1757–1767 (2002).
25. U Martínez-Peñas, On the similarities between generalized rank and Hamming weights and their applications to network coding. IEEE Trans. Inf. Theory **62**, 4081–4095 (2016)
26. MA Alsheikh, S Lin, D Niyato, HP Tan, Machine learning in wireless sensor networks: algorithms, strategies, and applications. IEEE Commun. Surv. Tutorials. **16**, 1996–2018 (2014)
27. N Nguyen, DL Jones, S Krishnamurthy, Netcompress: coupling network coding and compressed sensing for efficient data communication in wireless sensor networks. Signal. Proc. Syst., 356–361 (2010)
28. S Feizi, M Medard, A power efficient sensing/communication scheme: joint source-channel-network coding by using compressive sensing. Commun. Control. Comput., 1048–1054 (2011)
29. S Chen, M Wu, K Wang, Z Sun, W Lu, Combining network coding and compressed sensing for error correction in wireless sensor networks. Int. J. Commun. Syst. **28**, 1303–1315 (2015)
30. J Wright, Y Ma, Dense error correction via l1-minimization. IEEE Int. Conf. Acoust. Speech Signal. Proc. **56**(7), 3033–3036 (2009)
31. J Wright, AY Yang, A Ganesh, SS Sastry, Y Ma, Robust face recognition via sparse representation. IEEE Trans. Pattern Anal. Mach. Intell. **31**, 210 (2009)
32. A Ganesh, J Wright, X Li, EJ Candes, Y Ma, Dense error correction for low-rank matrices via principal component pursuit. IEEE Int. Symp. Inf. Theory Proc. **41**(3), 1513–1517 (2010)
33. C Qiu, N Vaswani, Recursive sparse recovery in large but correlated noise. Commun. Control. Comput. 752–759 (2011)
34. M Ali, M Kuijper, A parametric approach to list decoding of Reed-Solomon codes using interpolation. IEEE Trans. Inf. Theory **57**, 6718–6728 (2011)
35. DE Bergsagel, CC Sprague, SW Ross, List decoding Reed-Solomon, algebraic-geometric, and gabidulin subcodes up to the singleton bound. ACM Symp. Theory. Comput. **2**(11), 843–852 (2013)
36. V Guruswami, *List Decoding of Error-Correcting Codes*. (Yanjing university press, Beijing, 2004)
37. K Eritmen, M Keskinoz, Improving the performance of wireless sensor networks through optimized complex field network coding. IEEE Sensors J. **15**, 2934–2946 (2015)