Towards Optimal Error-Estimating Codes through the Lens of Fisher Information Analysis

Nan Hua
School of Computer Science
Georgia Institute of Technology
nanhua@cc.gatech.edu

Ashwin Lall
Department of Math and Computer Science
Denison University
lalla@denison.edu

Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
bli@eecg.toronto.edu

Jun (Jim) Xu
School of Computer Science
Georgia Institute of Technology
jx@cc.gatech.edu

ABSTRACT

Error estimating coding (EEC) has recently been established as an important tool to estimate bit error rates in the transmission of packets over wireless links, with a number of potential applications in wireless networks. In this paper, we present an in-depth study of error estimating codes through the lens of Fisher information analysis and find that the original EEC estimator fails to exploit the information contained in its code to the fullest extent. Motivated by this discovery, we design a new estimator for the original EEC algorithm, which significantly improves the estimation accuracy, and is empirically very close to the Cramer-Rao bound. Following this path, we generalize the EEC algorithm to a new *family* of algorithms called gEEC (generalized EEC). These algorithms can be tuned to hold 25-35% more information with the same overhead, and hence deliver even better estimation accuracy—close to optimal, as evidenced by the Cramer-Rao bound. Our theoretical analysis and assertions are supported by extensive experimental evaluation.

Categories and Subject Descriptors

E.4 [**Coding and Information Theory**]: Error control codes; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design - Wireless communication

General Terms

Algorithms, Theory

Keywords

Error Estimating Coding, Fisher Information

SIGMETRICS' 12, June 11–15, 2012, London, England, UK.

1. INTRODUCTION

Estimating the bit error rate (BER) in packets transmitted over wireless networks has been established as an important research problem in the seminal work of Chen *et al.* [3]. It was shown in [3] that, if the BER in packets can be accurately estimated, important operations in wireless networks such as packet re-scheduling, routing, and carrier selection can all be performed with greater efficiency. A simple and elegant technique, called error estimating codes (EEC), was hence proposed in [3] to estimate this BER. The basic idea of EEC is to have the transmitter send a small error estimating codeword along with each packet that will allow the number of bits in the packet flipped during the (wireless) transmission (i.e., the BER) to be inferred at the receiver. Using an EEC of $O(\log n)$ bits for a packet n bits long, their technique guarantees that the estimated BER falls within $1 \pm \epsilon$ of the actual BER with probability at least $1 - \delta$, where ϵ and δ are tunable parameters that can be made arbitrarily small at the cost of increased constant factor in $O(\log n)$, the coding overhead.

A natural question to ask is whether EEC achieves the best tradeoff between space ($O(\log n)$) and estimation accuracy ((ϵ, δ) guarantee) in solving the BER estimation problem. Our earlier work [6] has answered this question definitively. In [6], we prove that at least $\Omega(\log n)$ bits are needed to achieve an (ϵ, δ)-approximation and EEC is therefore asymptotically optimal. However, we also show that EEC has not achieved the optimal tradeoff down to the constant factor, by proposing a different coding algorithm, called the *Enhanced Tug-of-War* (EToW) sketch, that can achieve the same (ϵ, δ)-approximation with a code size 60% smaller than that used by EEC.

In this work, we follow up with a deeper and more important question of why EEC has not achieved the desired optimal space-accuracy tradeoff. Is it because its encoding algorithm has not packed as much information into the codewords as possible (i.e., inefficient encoding), or because its decoding algorithm has not made full use of information packed into these codewords (i.e., inefficient decoding), or both? This question is important because EEC is simpler and "cleaner" (to be explained shortly) than EToW, and if we can somehow bring its efficiency to be on par with or exceed that of EToW, it will likely be the preferred solution to the BER estimation problem. In an effort to thoroughly answer this question in this work, we have made the following two major contributions.

First, we demonstrate that EEC decoding is inefficient by deriv-

ing the amount of Fisher information contained in EEC codewords and showing that the variance of the estimator used in EEC encoding is much larger than the corresponding Cramer-Rao bound. We then propose a new estimator that achieves a significantly higher estimation accuracy and is provably near-optimal by almost matching the Cramer-Rao bound. Our experiments show that this new estimator allows us to reduce the coding overhead by as much as two to three times while achieving the same BER estimation accuracy. More importantly, EEC with this new estimator performs almost as well as EToW, with the same coding overhead. Another salient property of this new estimator is that its variance can be approximated by a closed-form formula, making it much easier to parameterize the EEC algorithm (i.e., to "tune") for optimal estimation accuracies (i.e., minimum variance) under various bit error models.

Second, we proceed to investigate whether there are inefficiencies with the encoding part of EEC. This question is, however, much harder to answer definitively because existing lower-bound techniques, rooted in the theory of communication complexity [7], only allow us to establish asymptotic space lower bounds such as the aforementioned $\Omega(\log n)$ bound. We try instead to compare the encoding efficiency, i.e., the amount of Fisher information per code bit, of EEC with that of EToW. It turns out that we got more than what we had bargained for. In an effort to derive the Fisher information formula of EToW, we have discovered that EEC and EToW can be viewed as different instances of a unified coding framework that we call generalized EEC (gEEC). In other words, gEEC can be parameterized into both EEC and EToW, and EEC can be viewed as a "degenerate" case of gEEC.

This generalization makes it easier for us to analyze and improve the designs of both EEC and EToW for two reasons. First, we need only design a single optimal decoder (i.e., estimator) for gEEC, which applies to both EEC and EToW, instead of one for each. This decoder is a Maximum Likelihood Estimator (MLE) with the Jeffreys prior, which in the case of EEC is the aforementioned estimator, and in the case of EToW performs better than a different estimator we developed for EToW in [6]. Second, the Fisher information formula derived for gEEC, which is in a closed form of matrix computations, applies to both EEC and EToW. However, we still derive the Fisher information formula of EEC separately in Section 3 because the derivations in this degenerate case are much simpler than those of gEEC, resulting in a closed-form formula (as the matrices degenerate into 1×1 scalars), and shed some insight on the roles that key parameters play in the formula that can get obscured in the general case. Through this unified framework of gEEC, we found that some parameterization of gEEC (similar to EToW, but not needing the extra error detection bits) can contain around 25% more information than the pure EEC scheme. This information gain cannot be fully decoded through EToW's decoder, but is achievable by gEEC's decoder.

The remainder of this paper is organized as follows. In Section 2 we discuss the related work most pertinent to this paper. In Section 3 we provide the Fisher information analysis of the original EEC algorithm. In Section 4, we propose the generalized EEC scheme and provide the corresponding analysis and estimator design. We evaluate the performance numerically as well as experimentally in Section 5. We give our conclusions in Section 6.

2. BACKGROUND AND RELATED WORK

In this section, we introduce some background on Fisher information and the Cramer-Rao bound, and briefly survey their applications in network measurement studies. We also briefly describe two prior approaches (EEC and EToW) for solving the BER estimation problem.

2.1 Fisher Information and Cramer-Rao Bound

In information theory and mathematical statistics, Fisher information quantifies the amount of information an observable parameterized random variable $X(\theta)$ carries about its unknown parameter θ. (Our description closely follows [4]; please consult it for more background on this topic.) In our context, the error estimating codewords and sketches correspond to random variable $X(\theta)$ and θ corresponds to the BER we would like to estimate in a received packet. The probability function of $X(\theta)$ takes the form $f(x;\theta)$. When θ is viewed as a constant and x as a variable, $f(x;\theta)$ is the probability density (or mass) of the random variable X conditional on the value of θ; When θ is viewed as a variable and x as a constant on the other hand, $f(x;\theta)$ is the likelihood function of θ, that is, the likelihood of the parameter taking value θ when the observed value of X is x. Fisher information of $X(\theta)$, which is a function of θ, is defined as

$$\mathbf{J}(\theta) \triangleq \mathbf{E}_\theta \left[\frac{\partial}{\partial \theta} \log f(X;\theta)) \right]^2. \quad (1)$$

Fisher information $\mathbf{J}(\theta)$ is an important quantity because it determines the minimum variance achievable by any unbiased estimator of θ given an observation of $X(\theta)$, through the Cramer-Rao lower bound (CRLB):

$$\mathbf{MSE}[\hat{\theta}] = \mathbf{Var}[\hat{\theta}] \geq \frac{1}{\mathbf{J}(\theta)}. \quad (2)$$

If a biased estimator of θ with bias $b(\theta)$ is used instead, we have a slightly different inequality:

$$\mathbf{MSE}[\hat{\theta}] \geq \frac{(1+b'(\theta))^2}{\mathbf{J}(\theta)} + b(\theta)^2. \quad (3)$$

While it is possible for a biased estimator to "beat" the Cramer-Rao bound for unbiased estimators (Formula (2)) when θ takes certain values (over which $b'(\theta)$ takes negative values), that biased estimator is not a clear winner since bias comes at a cost and may not be desirable to many applications.

In the context of this work, where the goal is to measure the "scale" of the bit error rate θ, the statistics of the relative error $\frac{\hat{\theta}-\theta}{\theta}$ and the log-difference $\log\hat{\theta} - \log\theta$ are more important. In particular, we prefer to use the statistics of the log-difference rather than the relative ratio to evaluate the performance of an estimator, since it assigns higher penalty to large deviations, making the comparison fairer for this application setting. For example, suppose the real value of θ is 0.1, and three large-deviation estimates are 0.05, 0.19 and 0.01, then the penalty for the last one will be much larger if measured by $\log\hat{\theta}$'s statistics. However, when measured by the relative error, the penalties of the latter two are the same and just around twice of the first.

The C-R bound for both $\frac{\hat{\theta}-\theta}{\theta}$ and $\log\hat{\theta} - \log\theta$ can be derived from (2) and the results are as follows:

$$\mathbf{Var}\left[\frac{\hat{\theta}-\theta}{\theta}\right] = \mathbf{Var}\left[\frac{\hat{\theta}}{\theta}\right] \geq \frac{1}{\theta^2 J(\theta)}. \quad (4)$$

$$\mathbf{Var}\left[\log\hat{\theta} - \log\theta\right] = \mathbf{Var}\left[\log\hat{\theta}\right] \geq \frac{1}{\theta^2 J(\theta)}. \quad (5)$$

Interestingly, they are bounded by the same value, $\theta^2 J(\theta)$. The second inequality (5) is derived by a transformation from (2). In other words, $\theta^2 J(\theta)$ is the Fisher information of $\log \theta$. We will use $\theta^2 J(\theta)$ frequently throughout the paper since it will directly determine the bound of the relative error/log difference.[1]

The maximum likelihood estimator (MLE) of θ, defined as $\hat{\theta}_{MLE} \triangleq \arg\max_\theta \{f(x;\theta)\}$, is known to be asymptotically normal (denoted as $N(*,*)$) under certain regularity conditions [8] and has the following distribution:

$$\hat{\theta}_{MLE} \sim N\left(\theta, \frac{1}{tJ(\theta)}\right), \tag{6}$$

where t here denotes the number of repeated independent experiments and $J(\theta)$ denotes the Fisher information contributed from each experiment. Hence the Cramer-Rao lower bound is (asymptotically) reached by the MLE.

Fisher information and Cramer-Rao bound analysis has been used in a few previous works on network measurement in the literature. It has been used by Ribeiro *et al.* [9] to derive the minimum number of samples needed for accurately estimating flow size distributions from outputs of a packet sampling process (e.g., sampled Cisco NetFlow). They also proposed an unbiased MLE estimator for this estimation problem that empirically matches the Cramer-Rao bound. This work was followed up in [10] by Tune *et al.* who demonstrated through Fisher information analysis that samples collected by flow sampling, which is much more expensive computationally, are more information-rich, in terms of Fisher information per bit, than packet sampling. They then proposed a new hybrid sampling technique called *dual sampling* that combines the advantages of both flow and packet sampling. Fisher information analysis is also used in recent work [11] to compare the information-richness of the samples collected by a few packet sampling and sketching techniques for the purpose of estimating flow size distributions.

2.2 Error Estimating Codes (EEC)

In EEC [3], the codeword for a packet consists of a set of $m = ab$ parity bits z_1, z_2, ..., z_m. They form a groups of size b each, $\{z_1, z_2, ..., z_b\}$, $\{z_{b+1}, z_{b+2}, ..., z_{2b}\}$, ..., $\{z_{(a-1)b+1}, z_{(a-1)b+2}, ..., z_{ab}\}$. Each parity bit z_i that belongs to group j (i.e., $(j-1)*b+1 \leq i \leq jb$) is calculated as the XOR of a set of $l_i = 2^j - 1$ bits uniformly (pseudo-)randomly sampled *with replacement* from the packet (viewed as a bit array). For example, in [3], a typical configuration of EEC scheme uses 9 groups (i.e., $a = 9$) of 32 parity bits (i.e., $b = 32$) each. In the following, we refer to each such group as a *level* to be consistent with the terms used in [3]. The codeword thus computed will be sent along with the packet to the receiver. Note the encoding scheme of EEC has some flavor of Low Density Parity Check (LDPC) codes although its parity check matrix is not strictly sparse as some of the rows can have as many as 2^a (including the parity check bit itself) ones in it, and in LDPC, no bit will be sampled more than once, which may happen in EEC due to its sampling with replacement nature.

Upon the receipt of a packet and its codeword (possibly with one or more bits flipped during transmission), the receiver will multiply them (viewed as a vector) by the same parity check matrix[2] and infer the BER from the outcome of this multiplication, which is often referred to as a *syndrome vector* in coding theory literature. From the syndrome vector, the inference algorithm (i.e., the decoder) used in [3] first decides on the group (i.e., level) of parity check bits that are expected to provide the best estimation accuracy. Then BER will be estimated *only* from the corresponding syndrome bits within that group. Such a commonsensical decoding procedure is however not optimal because other layers (groups) of parity bits, especially the neighboring layers, can provide much additional information to BER estimation, but are ignored by the decoder used in [3], as we will show shortly.

2.3 Enhanced Tug-of-war (EToW)

The Tug-of-war sketch (ToW) [1] is a sketch data structure (it can be viewed as an exotic type of error-correction code) originally proposed for estimating the L_2 norm of a single data stream. ToW can be extended for solving the problem of estimating BER, which is *almost* the same as that of measuring the Hamming distance and equivalently the L_2 norm between two binary vectors. A codeword in ToW, also referred to as a *sketch*, consists of a set of sub-sketches. In the context of this problem, in which all data items are binary, the value of each ToW sub-sketch is set to the number of 1's minus the number of 0's in the binary vector that results from XOR-ing the packet $\vec{b}$, viewed as a binary vector, bitwise with a pseudorandomly[3] generated binary vector. This bitwise exclusive-OR operation is mathematically equivalent to a projection of the vector $\vec{b}$ over a random direction (vector) $\vec{s}$, i.e. the inner product of $\vec{b}$ and $\vec{s}$. When comparing the Hamming distance between two binary data vector $\vec{b}$ and $\vec{b'}$, we can use the difference between their random projections $X = \vec{b} \cdot \vec{s} - \vec{b'} \cdot \vec{s}$ to estimate the norm of $\vec{b} - \vec{b'}$. Notice that the sketch (random projections) is much smaller in size ($O(log(n))$) than the length of original data vector (n) and hence is a succinct coding of data.

In the Enhanced ToW (EToW) scheme proposed in [6], this sketch is sent along with the packet to the receiver. Upon the receipt of the packet, which may have one or more bits flipped during transmission, the receiver will compute a new sketch from the packet received by projecting it over the same set of random directions (vectors). Then a decoder extended from the inference algorithm used in [1] will examine the differences between the sketch computed by the sender (and sent along with the packet) and the new sketch computed from the received packet, to arrive at a BER estimate.

The Enhanced ToW proposed in [6] extended ToW to make it appropriate and more space-efficient for BER estimation in the follow three ways. **First**, each sub-sketch in the original ToW scheme is encoded as a $\log_2(n)$-bit integer, where n is the maximum packet size in bits, so that no "overflow" can happen. However, since this difference (between the number of 1's and the number of 0's) is statistically much smaller than the maximum value n and in fact has mean 0, it makes sense to use "just enough" bits instead of $\log_2(n)$ to encode it. After all, these sub-sketches (differences) need to be sent along with the packet over wireless networks (the target network applications of error estimating codes) and in wireless communications, every bit counts! Therefore, in EToW, "barely enough" bits are allocated for each sub-sketch so that there is a very small probability (say 0.05) for overflows to happen. When an overflow happens, the most significant bit(s) are truncated. While this situation may sound disastrous to readers, it is in fact not that bad for two reasons: (1) most other sub-sketches have no overflows and can help correct the situation, and (2) the overflow situation most likely also occurs in the sub-sketch computed from the received packet and it is the difference between the sub-sketches

[1] There is another concept called relative Fisher information established in information theory which is not related to anything here.

[2] Both the sender and the receiver know this matrix.

[3] Both the sender and the receiver know how to generate the pseudorandom vectors. The pseudorandom vectors for computing different sub-sketches are mutually independent.

(likely both truncated) that counts. **Second**, the sketch sent along with the packet may itself be corrupted during transmission, which is a situation that ToW needs not worry about in its data streaming contexts. In EToW, lightweight error-detection codes are used to provide special protection to the sketch so that the receiver knows which sub-sketches are corrupted and can exclude them from BER estimation. **Third**, unlike in ToW, where bits are sampled with probability 1 (*without replacement*) to produce the random projection, in EToW, bits are sampled with probability less than 1 *with replacement* and only those sampled bits are used to produce the sub-sketch through the aforementioned random projection process. While such a sampling process was adopted in the EToW paper [6] purely for performance (i.e., BER estimation accuracy) considerations, it has the following two unintended (and perhaps desirable) consequences for this work: (1) While the Fisher information analysis of ToW is impossible due to its sampling without replacement nature, that of EToW is mathematically manageable. (2) EToW's adoption of sampling with replacement makes it possible for us to unify EToW with EEC, which samples bits also with replacement, through our gEEC (generalized EEC) framework, to be described in Section 4. With these three enhancements, EToW achieves 50-70% reduction in coding overhead, compared with an EEC scheme with similar BER estimation accuracy, as shown in [6].

3. FISHER INFORMATION ANALYSIS OF EEC

In this section, we analyze the Fisher information contained in an EEC codeword. Since each bit in an EEC codeword is generated in the same way independent of each other, it suffices to analyze the contribution from each bit. The Fisher information of the codeword is simply the sum of the Fisher information contained in each bit.

Throughout this paper, we will use notation $\check{z}_i$'s to denote the codeword bits (sent along with the packet) received (hence subjected to transmission errors), and use z_i''s to denote the codeword bits calculated from the packet received. In the EEC scheme, z_i and z_i' are the parity bits of the l_i bits on the same locations of the original and the received packets. The receiver computes their difference $X_i = \check{z}_i \oplus z_i'$, $i = 1, 2, \ldots, m$, and infers the error rate θ from the X_i's.

3.1 Fisher Information Contained in each EEC bit

To simplify the exposition, we first perform the Fisher information analysis under the unrealistic assumption that the codeword is immune from corruption during transmission. In this case, $\check{z}_i$'s, the codeword bits received, are the same as z_i's, the codeword bits sent. We will then handle the more realistic case, without immunity, at the end of this section.

3.1.1 The case with "immunity"

Recall X_i defined above indicates whether or not the i_{th} parity equation holds. In the case with immunity, the likelihood function of observing $X_i = 1$ (i.e., odd number of bits "flipped" during transmission among the set of bits sampled) is as follows:

$$f(X_i = 1, \theta) = \Pr(X_i = 1|\theta) \quad (7)$$

$$= \sum_{j=0}^{2j+1 \le l_i} \binom{l_i}{2j+1} \theta^{2j+1} (1-\theta)^{l_i - 2j - 1}$$

$$= \frac{1 - (1-2\theta)^{l_i}}{2}. \quad (8)$$

The Fisher information contained in each X_i can be calculated as follows:

$$J_i^{EEC}(\theta) = \mathbf{E}_\theta[(\frac{\partial}{\partial\theta} \log f(X_i;\theta))^2] \quad (9)$$

$$= \Pr(X_i = 1|\theta)(\frac{\partial}{\partial\theta} \log f(X_i = 1;\theta))^2 + \Pr(X_i = 0|\theta)(\frac{\partial}{\partial\theta} \log f(X_i = 0;\theta))^2$$

$$= \frac{1-(1-2\theta)^{l_i}}{2} \left(\frac{2l_i(1-2\theta)^{l_i-1}}{1-(1-2\theta)^{l_i}}\right)^2 + \frac{1+(1-2\theta)^{l_i}}{2} \left(-\frac{2l_i(1-2\theta)^{l_i-1}}{1+(1-2\theta)^{l_i}}\right)^2$$

$$= \frac{4l_i^2(1-2\theta)^{2l_i-2}}{1-(1-2\theta)^{2l_i}}. \quad (10)$$

Before we aggregate Fisher information contributed by all the codeword bits, we would like to highlight some nice properties of the derived Fisher information for a single bit (10). For convenience of comparisons, we define the Fisher information (of $\log\theta$) of each bit as $\eta(l_i, \theta)$:

$$\theta^2 J_i^{EEC}(\theta) = \frac{4\theta^2 l_i^2 (1-2\theta)^{2l_i-2}}{1-(1-2\theta)^{2l_i}} \quad (11)$$

$$\triangleq \eta(l_i, \theta). \quad (12)$$

As we explained earlier, this is inversely proportional to the Cramer-Rao bound of the relative variance. The larger $\eta(l, \theta)$, the tighter the bound of the relative error (and the log-difference) that can be achieved.

Values of $\eta(l_i, \theta)$ for various l_i (number of bits sampled) values are plotted in Figure 1. We can see that the $\eta(l, \theta)$ curves corresponding to different l values are actually very similar in shape to each other. A larger parity group size l is better for estimating smaller θ values and vice versa. These curves also have similar "heights" except when l gets really small (targeting extremely high BER close to the maximum possible value of 0.5). This means that the "peak estimation powers" of different parity bits are about the same. The maximum (i.e., "height") of each curve is always reached around $\theta = 0.4/l$, which means the parity bit computed from l sampled bits yields the best estimation for θ when θ is around $0.4/l$.

We can even quantify how much "information about θ" flows into the whole estimation spectrum by the following integral formula, which is the area covered by the FI(rel) curves in Figure 1:

$$\mathbf{S^J}(l) = \int_0^{0.5} \theta^2 J(l, \theta) d\log\theta \quad (13)$$

In (13), the upper limit of the integral is 0.5 is because all formulas derived above (starting from (8)) are only valid when θ is less than or equal to 0.5.

From a practical perspective, an integral over the full spectrum might not be too useful since practical applications might be only interested in a particular range of spectrum rather than a full spectrum, such as $[10^{-3}, 0.15]$ proposed in [3]. Moreover, maintaining accuracy over a certain threshold might be even more practical than an integral. However, we find $\mathbf{S^J}$ is still a very good indicator of performance since it remains almost the same for EEC bits with different parameter l, which is not surprising since the Fisher information curves are similar to each other in shape and hence the "area under" the curves are also close to one another. In Figure 1, we list the values of $\mathbf{S^J}(l)$ of different l's in the legend, where constant $C = \frac{1}{24}\pi^2 \approx 0.4112$, which is a provable limit of $\mathbf{S^J}(l)$ when l goes infinity. When l increases, $\mathbf{S^J}(l)$ gets closer to C. In other

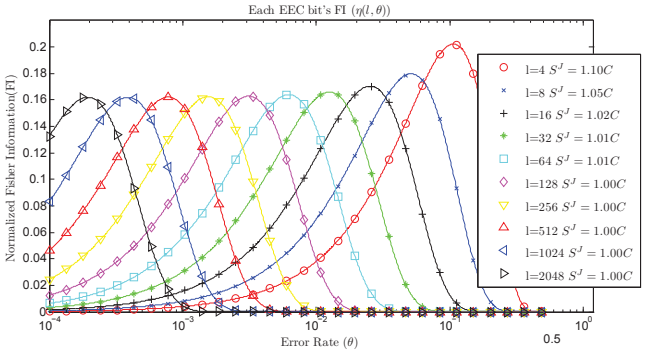

Figure 1: Fisher Information (of $\log\theta$) of each EEC bit in function of l and θ

words, the total information $\mathbf{S^J}$ contributed by each bit is all on the same scale and almost invariant to parameter l. In Section 5.1, we will see that this criteria $\mathbf{S^J}$ helps us to differentiate the strengths of different schemes and guides us in selecting better schemes.

3.1.2 The case without "immunity"

We proceed to perform the Fisher information analysis of EEC when the codeword sent along with the packet is no longer assumed to be immune from corruptions during transmission. The amount of Fisher information contained in each parity bit X_i can be derived as in (14). We omit the details of this derivation since it is a special case of the Fisher information analysis (without "immunity") in the gEEC framework in Section 4.2. Note that, unlike the case with "immunity" where the analysis rigorously holds, the analysis for the case without "immunity" only rigorously holds when i.i.d. random binary errors are assumed or random placement of bits are assumed.

$$\theta^2 J_i^{EEC}(\theta) = \frac{4(l_i+1)^2(1-2\theta)^{2l_i}}{1-(1-2\theta)^{2l_i+2}} \quad (14)$$

Notice that the RHS of (14) is equal to $\eta(l_i+1,\theta)$, only slightly different from (10).

3.2 Combining the contributions of the different levels of bits

Since all parity bit are calculated from independent samples with replacement and hence each of them is independent of every other, the Fisher information of the codeword is simply the sum of their Fisher information together:

$$J^{EEC}(\theta) = \sum_{i=1}^{m} J_i^{EEC}(\theta). \quad (15)$$

In Figure 2, we plot the Cramer-Rao bound of the EEC scheme with the typical configuration (9 levels, 32 bits per level) and we also plot the Cramer-Rao bound of schemes with only one level of 32 bits. From Figure 2, we can see that after using the information of all levels, the estimation accuracy has the potential to be considerably improved. In other words, suppose there are two optimal estimators, the first can use only the information contained in one level of bits, while the second can use the information contained in all levels. Then the variance of the first estimator will be four times as large as compared to the second for most θ values. This means the codeword size has to be four times as large for the first estimator to match the second in estimation accuracy.

In the design of the original estimator in [3], they first identify the level of bits likely to be the most accurate for estimating θ and then only use that level of bits for estimation. Hence only one level of information is used in the final estimate. In Section 3.5 of the technical report [2], the authors have proposed an improved estimator that can make use of two neighboring levels of parity bits. In Figure 2, we use $\hat{\theta}_1$ to denote the original one proposed in [3] and use $\hat{\theta}_2$ to denote the improved version proposed in [2]. We can see that neither estimator is close to the Cramer-Rao bound corresponding to the amount of Fisher information contained in such one or two levels of bits (In other words, their estimators have not made full use of even the information contained in such one or two levels of bits) and far from the Cramer-Rao bound corresponding to the Fisher information of all the codeword bits. We note that when θ is larger than 0.15, the relative MSE falls below the Cramer-Rao bound; This is because the estimator is actually very biased in that region. According to (3), this might lead to smaller MSE.

3.3 MLE estimator for EEC scheme

In this section, we present our MLE decoder that matches the Cramer-Rao bound, shown in the following formula.

$$\hat{\theta}_{MLE} = \arg\max_{\theta}\{\sum_{i=1}^{m}\log\Pr(X_i|\theta)\} \quad (16)$$

$$= \arg\max_{\theta}\{\sum_{i=1}^{m}\log\frac{1-(1-2\theta)^{l_i}}{1+(1-2\theta)^{l_i}}X_i \quad (17)$$

$$+\log(1+(1-2\theta)^{l_i})\} \quad (18)$$

It has been discovered in [5] that the non-informative prior, Jefferys invariant prior, can remove the $O(\frac{1}{N})$ component in bias for the family of exponential models. Here, although the likelihood function is not a closed-form distribution, we find that Jeffreys prior will help the MLE estimator to achieve better results. The MLE with the Jefferys prior, denoted as $\hat{\theta}_{MLE_j}$, is

$$\hat{\theta}_{MLE_j} = \arg\max_{\theta}\{\sum_{i=1}^{m}\log\Pr(X_i|\theta)+\frac{1}{2}\log J(\theta)\}$$

$$= \arg\max_{\theta}\{\sum_{i=1}^{m}\log\frac{1-(1-2\theta)^{l_i}}{1+(1-2\theta)^{l_i}}(X_i-\frac{1}{2})$$

$$+\log(2l_i(1-2\theta)^{l_i-1})\}.$$

Here $p(\theta)\sim\sqrt{J(\theta)}$, where $p(\theta)$ is the a priori distribution of the parameter θ and $J(\theta)$ is the Fisher information calculated in (15). Note that the MLE with Jefferys prior will still asymptotically reach the Cramer-Rao bound.

4. DESIGN AND FISHER INFORMATION ANALYSIS OF GENERALIZED EEC (GEEC)

In the previous section, we showed through Fisher information analysis that the original decoder for EEC used in [3] is far from optimal and our new decoder is near-optimal (by almost matching the Cramer-Rao bound) given the amount of Fisher information contained in an EEC codeword. Now we would like to find out whether the EEC encoding scheme is efficient enough by comparing its Fisher information (per bit) with that of EToW. How-

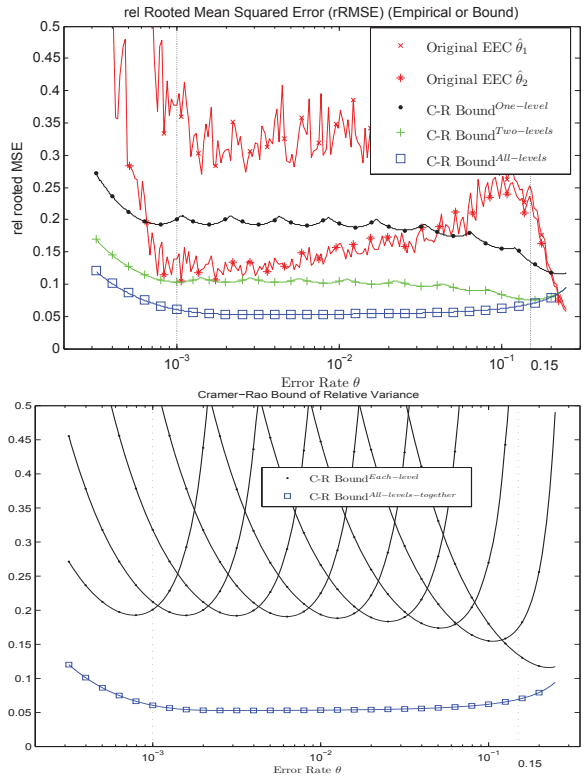

Figure 2: the empirical performance of EEC's original estimators and the associated Cramer-Rao Bound (all levels, each level, and the envelope of only one level/two levels). The EEC scheme is composed by nine levels and 32 bits each level.

ever, when we were performing the Fisher information analysis of EToW, we discovered a generalized EEC (gEEC) scheme that can be parameterized into both EEC and EToW. Fisher information analysis of EToW can thus be generalized to that of gEEC so we shift our "target" for comparison to gEEC. Through this unified framework of gEEC, we have found that some parameterizations of the gEEC family contain 25% or more Fisher information per bit in their codewords than EEC. In other words, the EEC encoding scheme is not very efficient either. The discovery of gEEC is important also for another reason: We have discovered a unified decoder (estimator) for gEEC that is near-optimal (by almost matching the Cramer-Rao bound) and when parameterized into EToW, is more accurate than the original EToW decoder proposed in [6]. Strictly speaking, it is not the clear winner however since the original EToW decoder has much lower computational and storage complexities.

The rest of the section is organized as follows. In Section 4.1, we briefly describe the gEEC encoding scheme by highlighting its differences from and connections to both EToW and EEC. In Section 4.2, we proceed to perform the Fisher information analysis of its codeword and design its estimator. We will see the gEEC decoder that almost matches the Cramer-Rao bound in Section 5.2.

4.1 The gEEC Encoding

A gEEC codeword (called *sketch*) of a packet (viewed as binary vector $\vec{b}$) consists of m sub-sketches $z_1, z_2, \ldots, z_m$. The value of sub-sketch z_i is set to the number of 1's contained in the binary vector generated by sampling l_i bits from the packet *with replacement*, which we refer to as $\vec{b_i}$, and then XOR-ing it with a pseudo random vector $\vec{s_i}$ bitwise. This operation is the same as in EToW, except that in EToW the value of z_i is set to the half of the difference between the number of 1's and the number of 0's. It can be shown that these two types of encodings can be made equally efficient and statistically equivalent with respect to the truncation operations (described shortly) with proper parameterizations.

The main difference between EToW and gEEC is that all $l_i's$ have to take the same value in EToW. This is not an artificially crafted difference because EToW's decoder, inherited from ToW and based on the method of moments, imposes this equal length requirement, while the new decoder we propose for gEEC that we will describe in Section 4.2.2, does not have such a requirement due to its MLE nature. Like in EToW, we use barely enough bits to encode each sub-sketch and "overflows" are handled in the same way through truncation. Moreover, different from EToW which needs extra checking bits to detect corruption inside the sketch, we will see that the estimators provided the gEEC's framework have built-in capability to decode the sketches which might suffer corruption in the case without "immunity".

The precise definition of z_i in gEEC scheme is as follows:

$$z_i = \sum_{j=1}^{l_i} (b_{i,j} \oplus \frac{1+s_{i,j}}{2}) \pmod{K_i}, \qquad (19)$$

where $K_i = 2^{k_i}$, k_i is the number of bits allocated to sub-sketch z_i. Each $s_{i,j}$ is a pre-computed pseudo-random number uniformly and independently selected from $\{-1, 1\}$, the same as the definition in the tug-of-war sketch [1, 6]. The function $\frac{1+s_{i,j}}{2}$ maps $s_{i,j}$ from $\{-1, 1\}$ to $\{0, 1\}$.

As shown below, the nature of the definition above is a random projection of $\vec{s_i}$ (only different from the $\vec{b_i} \cdot \vec{s_i}$ by a pseudo-random constant, i.e. a number that is the same in both sender and receiver):

$$z_i = \sum_{j=1}^{l_i} (b_{i,j} \oplus \frac{1+s_{i,j}}{2}) \qquad (20)$$

$$= \sum_{j=1}^{l_i} \frac{1+(2b_{i,j}-1)s_{i,j}}{2} \qquad (21)$$

$$= \vec{b_i} \cdot \vec{s_i} - \frac{1}{2}\vec{1} \cdot \vec{s_i} + \frac{l_i}{2}. \qquad (22)$$

It can be shown that when we allocate only 1 bit for each sub-sketch, the sketch becomes a parity array and in this case gEEC "degenerates" into a scheme statistically equivalent to EEC with the same l_i values. We can also see that gEEC becomes statistically equivalent to EToW without sketch protection (explained next) when $l_i's$ are set to the same value.

4.2 Fisher information analysis

In this section, we analyze the Fisher information contained in each gEEC codeword. Recall from Sec. 2.3 that in EToW we need to protect the sketch against corruptions during transmission using lightweight error-detection codes. We will show no such protection is needed in gEEC because, unlike the EToW decoder, the proposed MLE decoder for gEEC is robust against such corruptions. In order to simplify the presentation of the analysis, we first analyze the Fisher information of a gEEC codeword assuming that the codeword is immune from corruption during transmission in Section 4.2.1 and then show how to remove this assumption in Section 4.2.2.

4.2.1 The case with "immunity"

Like in Section 3, we use $\vec{b_i'}$ to denote the set of bits sampled *with replacement* from the packet received (subject to corruptions during transmission) that are used to compute the i_{th} sub-sketch z'_i at the receiver side, and use $\vec{b_i}$ to denote the corresponding set of bits sampled from the packet sent (can be different from $\vec{b_i'}$ due to corruptions during transmission) that are used to compute the i_{th} sub-sketch z_i at the receiver side. In this section, we derive the Fisher information of the sketch under the aforementioned immunity assumption that these sub-sketches z_1, z_2, ..., z_m will arrive at the receiver without having any of their bits corrupted during transmission.

The receiver can calculate the difference X_i between z'_i and z_i

$$X_i \triangleq z'_i - z_i (\text{mod } K), i \in [m] \tag{23}$$

Here $K = 2^k$ where k is the number of bits we allocate to each sub-sketch. The effect of the aforementioned (possible) overflow and resulting truncation is reflected in "modulo K". It can be shown that this observation X_i is the following function of the error vector $\vec{e_i}$ (the difference vector between $\vec{b_i}$ and $\vec{b_i'}$), where s_i is the aforementioned pseudorandom vector over which the sampled bit vectors $\vec{b_i}$ and $\vec{b_i'}$ are linearly projected:

$$\begin{aligned} X_i &= \vec{b_i} \cdot \vec{s_i} - \vec{b'}_i \cdot \vec{s_i} (\text{mod } K), \text{due to (22)} && (24)\\ &= \vec{e_i} \cdot \vec{s_i} (\text{mod } K). && (25) \end{aligned}$$

Since observations X_1, X_2, ..., X_m are independent random variables, the likelihood function of the random vector $\langle X_1, X_2, ..., X_m \rangle$ is the product of the likelihood functions of these random variables. The likelihood function of X_i (for arbitrary i) can be derived as follows.

For convenience, we drop the subscript from X_i and denote it simply as X. In the following we will derive the probability mass function (PMF) of X, which takes values from the set of $K = 2^k$ integers $\{0, 1, ..., K-1\}$. It can be shown that its PMF can be determined by a K-dimensional vector $\vec{\gamma}(\theta, l) \equiv \langle \gamma_0(\theta, l), \gamma_1(\theta, l), ..., \gamma_K(\theta, l) \rangle$ where $\gamma_i(\theta, l) \equiv \Pr(X = i | \theta, l)$, $i \in [0, 1, ..., K-1]$. Note that each scalar γ_i is a function of the error rate θ and the number of bits sampled l (here the subscript i is dropped from l_i). We show that $\vec{\gamma}(\theta, l)$ can be computed from the following recurrence relation.

LEMMA 1.

$$\vec{\gamma}(\theta, l)_{1\times K} = \vec{\gamma}(\theta, l-1)_{1\times K} \mathbf{M}(\theta)_{K\times K}, \tag{26}$$

where

$$\mathbf{M}(\theta) = \begin{bmatrix} 1\text{-}\theta & \theta/2 & & \cdots 0 \cdots & & & \theta/2 \\ \theta/2 & 1\text{-}\theta & \theta/2 & & \cdots 0 \cdots & & \\ & \theta/2 & 1\text{-}\theta & \theta/2 & \cdots 0 \cdots & & \\ & & \cdots & & \cdots & & \\ & \cdots 0 \cdots & & & \theta/2 & 1\text{-}\theta & \theta/2 \\ \theta/2 & & \cdots 0 \cdots & & & \theta/2 & 1\text{-}\theta \end{bmatrix}_{K\times K} \tag{27}$$

The initial condition for the above is

$$\vec{\gamma}(\theta, 0) = [1, 0 \cdots, 0]_{1\times K}. \tag{28}$$

PROOF. Let $\vec{e} = \langle e_1, e_2, \cdots, e_l \rangle$ and $\vec{s} = \langle s_1, s_2, \cdots, s_l \rangle$. We define the following interim random variables Y_j, $j = 0, \cdots, l$:

$$Y_j = \sum_{k=1}^{j} e_k s_k. \tag{29}$$

Clearly, $X = Y_l (\text{mod } K)$. Due to the sampling with replacement policy, the increment in each step $\Delta Y_j = Y_{j+1} - Y_j = e_{j+1} s_{j+1}$ is independent of every other. Hence the random variables $\{Y_j\}_{0\le j\le l}$ make up a Markov chain. In each step, with probability $1-\theta$ an unchanged bit is selected and hence $\Delta Y_j = 0$; with probability θ a changed bit is selected, and half of these increments are $+1$ and the other half -1 since s_k is uniformly at random from $\{-1, 1\}$.

Hence the distribution of ΔY_j is:

$$\Delta Y_j = \begin{cases} -1 & \text{with prob } \theta/2, \\ 0 & \text{with prob } 1-\theta, \\ 1 & \text{with prob } \theta/2. \end{cases} \tag{30}$$

Mapping $\{Y_j\}_{0\le j\le l}$ into the finite field Z_K, formula (30) becomes the transition matrix, which is the circular matrix M defined in (27). □

To allow for efficient matrix computation, $\mathbf{M}(\theta)$ can be diagonalized as follows.

$$\mathbf{M}(\theta) = \frac{1}{K} \mathbf{\Omega}' Diag(d_0, d_1, \ldots, d_{K-1}) \mathbf{\Omega} \tag{31}$$

where $\Omega = \{\omega_{ik}\}$, $\omega_{ik} = \exp(\frac{2\pi i k j}{K})$, j is the imaginary unit, and $d_i = 1 - \alpha_i \theta$, $\alpha_i = 2\sin^2(i\pi/K)$, which is actually the Fourier transform matrix.

Considering that $\vec{\gamma}(\theta, 0) = [1, 0 \cdots, 0]_{1\times K}$, we have

$$\begin{aligned} \vec{\gamma}(\theta, l) &= [1, 0, \cdots, 0] \mathbf{M}(\theta)^l \\ &= \frac{1}{K} [1, 0, \cdots, 0] \mathbf{\Omega}' Diag(d_0^l, d_1^l, \ldots, d_{K-1}^l) \mathbf{\Omega} \\ &= \frac{1}{K} [d_0^l, d_1^l, \ldots, d_{K-1}^l] \mathbf{\Omega}. && (32) \end{aligned}$$

The Fisher information of the gEEC sketch can be calculated as follows:

$$\begin{aligned} J(\theta, l) &= E_\theta (\frac{\partial}{\partial \theta} \log f(X_i; \theta))^2 \\ &= \sum_{j=0}^{K-1} \gamma_j(\theta) (\frac{\mathrm{d}}{\mathrm{d}\theta} \log(\gamma_j(\theta)))^2 \\ &= \sum_{j=0}^{K-1} \frac{l^2 \{[0, \alpha_1 d_1^{l-1}, \ldots, \alpha_K d_{K-1}^{l-1}] \mathbf{\Omega}\}_j^2}{K \{[d_0^l, d_1^l, \ldots, d_{K-1}^l] \mathbf{\Omega}\}_j}, && (33) \end{aligned}$$

where γ_j denotes the j^{th} item in vector $\vec{\gamma}$ and $\{\vec{v}\}_j$ denotes the j^{th} item in vector $\vec{v}$.

It can be shown that when we set k to 1 (so that gEEC degenerates into EEC), formula (33), the Fisher information of gEEC codeword is equal to formula (10), that of EEC.

4.2.2 The case without immunity

In the previous section, we have performed an Fisher information analysis of gEEC under the assumption that the codewords (sketches) sent along with the packets are not subject to corruptions during transmission (i.e., "with immunity"). In reality, these codewords are certainly not immune to bit errors. In this section, we perform the Fisher information anaysis without this "immunity assumption".

Suppose the sender sends out a sub-sketch z_i and the receiver receives $\check{z}_i$. Now $\check{z}_i$ may differ from z_i as the "immunity" has been taken away. Having no knowledge of z_i, the receiver has to use $\check{z}_i$ and $z_i{}'$ computed from the received packet to infer the bit error rate θ. The conditional (upon θ) joint probability mass function of $\langle \check{z}_i, z_i{}' \rangle$ is shown as follows. For ease of notation, we remove the subscript i from the notations and get:

$$\Pr(\check{z}, z'|\theta) \propto \sum_{z=0}^{K-1} \Pr(z, \check{z}|\theta) \Pr(z'|\theta, z)$$
$$\propto \sum_{z=0}^{K-1} \Pr(z) \Pr(\check{z}|\theta, z) \Pr(z'|\theta, z). \quad (34)$$

Note that the formula (34) above only holds under the assumption that the binary error flips on sketch bits and the error flips on data bits are independent, i.e., $\check{z} \perp z'|\theta, z$, which requires either that the binary errors are i.i.d. distributed inside the sketch or that the locations of all bits participating including the sketch bits are all sampled with replacement from the packet, which is not possible to be strictly guaranteed in reality. Hence, we readily admit that, different from the analysis for the case with "immunity" which is rigorously held, the analysis for the case without "immunity" does not model reality perfectly, though we believe that it should be very close.

The matrix representation of (34) is as follows:

$$\mathbf{P}(\theta)_{K\times K}^{final} \propto \mathbf{\Lambda}\mathbf{M}(\theta)^l \mathbf{T}(\theta)_{K\times K}, \quad (35)$$

In (35), matrix $\mathbf{M}$ corresponds to $\Pr(z'|\theta, z)$ and can be calculated using (31). Matrix $\mathbf{\Lambda}$ is a diagonal matrix and corresponds to $\Pr(z)$. Its diagonal elements can also be calculated using (32) with $\theta = 1$. We acknowledge that an ulterior motive for us to define z_i as the number of 1's, rather than the number of 1's minus the number of 0's, in the bitwise-XOR of $\vec{b_i}$ and $\vec{s_i}$, is that it makes these formulae much "cleaner".

Matrix $\mathbf{T}$ is the transition matrix that corresponds to $\Pr(\check{z}, \check{q}|\theta, z)$. Each of its entry T_{ij} is defined as follows:

$$\mathbf{T}_{i,j} = \theta^{d_{ij}}(1-\theta)^{k-d_{ij}}, \quad (36)$$

where d_{ij} is the Hamming distance between the binary representation of i and j.

All entries in (35) are differentiable and the Fisher information can be derived in a way similar to (33). In the interest of space the final Fisher information formula is omitted here.

Note that when $k = 1$, gEEC degenerates to EEC, and the final Fisher information formula can be shown to be equivalent to EEC's Fisher information formula (14).

4.3 Our MLE Estimators

In this section, we derive the MLE decoder for gEEC that perform much better than both EEC and EToW decoders, as we will show in Sec. 5.2. Again, we first derive it for the easier case with "immunity" and then proceed to take the "immunity" away.

In the case with "immunity", our observations are X_i, which is the difference between each pair of $\check{z}_i$ and z'_i, $i = 1, 2, ..., m$, and our goal is to estimate θ. The maximum likelihood estimator is

$$\hat{\theta}_{MLE} = \arg\max_{\theta}\{\sum_{i=1}^{m} \log\{\vec{\gamma}(\theta, l_i)\}_{X_i}\} \quad (37)$$

Here $\{\vec{\gamma}(\theta, l_i)\}_{X_i}$ denotes the X_i^{th} scalar in $\vec{\gamma}$ and $\vec{\gamma}(\theta, l)$ and $J(\theta, l)$ have both been derived earlier. The MLE estimator with Jeffreys prior (MLE-J) is

$$\hat{\theta}_{MLE-J} = \arg\max_{\theta}\{\frac{1}{2}\log J(\theta) + \sum_{i=1}^{m} \log\{\vec{\gamma}(\theta, l_i)\}_{X_i}\} \quad (38)$$

We introduce $MLE-J$ because it performs better than the plain vanilla MLE empirically, which will show in Section 5.2.

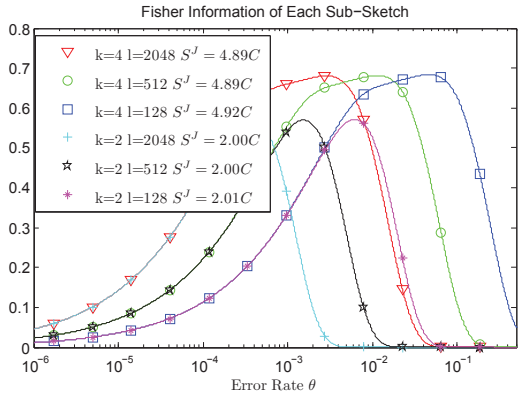

Figure 3: gEEC's Fisher information: Relationship to l and k in the case with immunity assumption

Similar to (38), the MLE estimator with Jeffreys prior for the case without immunity is

$$\hat{\theta} = \arg\max_{\theta}\left\{\frac{1}{2}\log J(\theta) + \sum_{i=1}^{m} \log\{\mathbf{P}(\theta, l_i)\}_{\check{z}_i, z'_i}\right\}, \quad (39)$$

where $\{\mathbf{P}(\theta, l_i)\}_{\check{z}_i, z'_i}$ denotes the $(\check{z}_i, z'_i)^{\text{th}}$ entry in the probability matrix $\mathbf{P}$.

This estimator will also be evaluated empirically in Section 5.2.

5. NUMERICAL AND EMPIRICAL RESULTS

In this section, we present an extensive array of numerical results for gEEC's Fisher information, as well as an empirical evaluation of gEEC's estimators with different configurations. Note that both EEC and EToW are just two of the sub-families in the gEEC family.

5.1 Fisher Information contained in the gEEC Family

Fisher information contained in each gEEC's sub-sketch is determined by two factors: the sampling group size l and the binary width k. For convenience, this is denoted as gEEC(l, k), and a sketch composed of m such sub-sketches is denoted by gEEC(m, l, k).

In Figure 3, we show the impact of different l's and k's in the case with immunity. We observe that, similar to Figure 1, the parameter l only "shifts" the curve. The larger l, the more resolution on lower θ's, while the total amount of "information flow" S^J remains almost the same. The parameter k, on the other hand, expands the "span" of the curve, leading to a wider spread of the estimation power on the spectrum.

To compare the total amount of "information flow" of different parameterizations more conveniently, we also list the value S^J—the area covered by each curve as defined in (13)—of each parameterization in the legend of the figure, where the constant C is $\frac{\pi^2}{24} \approx 0.4112$, the lower limit of the area covered by the EEC bit's Fisher information curve, as discussed in Section 3. Based on the S^J values in Figure 3, we conclude that $k = 2$ is not able to bring any additional benefit. In contrast, one sub-sketch with $k = 4$ can gain more information over 4 independently coded EEC bits.

The next question we ask is: How much information is lost due to the contamination of the sketch? To measure this, we plot the Fisher information curve with $k = 4$ or 6 and $l = 2048$, 512 or

	Case with immunity			Case without immunity		
k	l=128	l=512	l=2048	l=128	l=512	l=2048
2	1.00	1.00	1.00	1.00	1.00	1.00
3	1.12	1.12	1.11	1.08	1.10	1.11
4	1.23	1.22	1.22	1.13	1.18	1.21
5	1.24	1.31	1.31	1.16	1.20	1.26
6	1.05	1.30	1.37	1.19	1.21	1.27
7	0.90	1.14	1.36	1.22	1.24	1.27

Table 1: The total information gain $\frac{S^J}{kC}$ ($C = \frac{1}{24}\pi^2$)

128 in Figure 4. We observe that the Fisher information curve is impacted only when θ is not small and the loss of information is also modest. For some cases, such as $k = 6$ and $l = 128$, the total information is improved even when the sketch is subject to errors.

We summarize the impact of l and k's on S^J by listing the $\frac{1}{kC}S^J$ values of different parameterizations in Table 1. From the perspective of total information gain S^J, more bits per sub-sketch usually improves the gain. We see that $k = 5$ gives around 25% more information per bit in the contaminated case, which can be translated to a similar ratio of reduction of the sketch size to achieve the same variance bound. It can be also shown that the performance of one sub-sketch with $k = 5$ can dominate six one-bit sub-sketches together.

However, a larger k is not always better. The larger k gets, the wider the span of the resolution curve gets, which might cover more range than needed. If l is small (for better resolution on large θ's), k should not need to be too large, otherwise it will be wasteful (such as the $k = 6$ and $l = 128$ case in the table). For a typical EEC application in wireless communications where the primary target parameter range is $[10^{-3}, 0.15]$, the span of $k = 5$ (or 6) would be sufficient. Moreover, a large k will mean a higher implementation cost for a relatively modest gain, which will be discussed soon.

5.2 Empirical Evaluation of Estimators

We now present an empirical evaluation of gEEC's estimators. We aim to use the figures in Figures 5,6,7 to answer two questions: (1) How far is the estimator's real performance from the theoretical results? and (2) How does this compare with previously proposed solutions? Each value in igures 5,6,7 is obtained with 1000 runs in our simulations, in the case without immunity.

The comparison metrics that we use are the relative mean squared error (rMSE, defined as $\frac{1}{\theta}\overline{(\hat{\theta}-\theta)^2}$), the mean squared error of $\log\hat{\theta}$ (defined as $\overline{(\log\hat{\theta}-\log\theta)^2}$)), the ratio of large errors (the ratio of $\hat{\theta}$ that are larger than 2θ or smaller than $\theta/2$), and the relative bias ($\overline{\frac{\hat{\theta}}{\theta}-1}$).As discussed in Section 2.1, although the Cramer-Rao lower bound for the relative MSE of $\hat{\theta}$ (4) and the MSE of $\log\hat{\theta}$ (5) are the same, we prefer to use the statistics of $\log\hat{\theta}$ for comparison since it allocates larger penalty to large deviations.

In Figure 5, we compare the performance of the original 288-bit EEC scheme with original estimators with two gEEC configurations, gEEC$(16, 768, 5)$ and gEEC$(16, 768, 6)$, whose total transmission costs are 80 and 96 bits, respectively. We see that our new sketches perform very well in $[0.001, 0.15]$ with much less transmission overhead than the original EEC, while the performance of the original EEC's estimator varies, especially when measured by harsher criteria such as the MSE of $\log\hat{\theta}$ and the ratio of large errors.

In Figure 6, we compare the performance of the four estimators of two schemes, the original EEC and gEEC$(56, 512, 5)$. All four estimators are derived from the newly proposed gEEC framework, two of which use the Jeffreys prior and the other two do not. We can see that the estimators with Jeffreys prior are generally better in the range where θ is relatively large and the inherent resolution of the scheme is relatively weak, no matter if measured by MSE or by bias. Since estimators with Jeffereys prior are usually empirically better, we always use that version in other comparisons.

In Figure 7, we compare the performance of four schemes, the original EEC, gEEC$(56, 512, 5)$, gEEC$(56, 768, 6)$ and EToW$(56, 768)$ with 5-bits per sub-sketch and 1-bit for detecting corruption, the transmission cost of which are almost the same. The first three use gEEC's MLE estimator with Jeffreys prior, while the last one uses EToW's moment-based estimator. Comparing these figures with Figure 5, we can see that, with the same transmission cost, the estimation accuracy is substantially improved. Moreover, we can see that the gEEC$(56, 512, 5)$ and gEEC$(56, 768, 6)$'s performance are generally better than the others. In a wide range they can achieve around a 30-50% reduction of MSE, compared with the EEC scheme with our new estimator. This is not surprising since we have already seen that a larger k can bring a modest improvement of estimation accuracy. We also observe that EToW's performance is only slightly worse, except when θ is close to or larger than 0.1.

Comparing the curves in Figure 5 and Figure 6, we can see that when the total number of sketches is large, the MLE estimator's performance is nearly equal to the Cramer-Rao lower bound, and its bias is also reduced as seen by comparing Figure 5(d) and Figure 5(d). Note that sometime the MSE of $\log\theta$ or rMSE might be below the Cramer-Rao bound when bias is high, which is possible since all curves of Cramer-Rao bound presented in the figures are for the unbiased estimator.

To summarize our comparisons above, our estimators, especially the ones with Jeffreys prior, can almost achieve the Cramer-Rao bound empirically. On one hand, gEEC$(16, 768, 5)$ (which is also EToW's scheme, just without EToW's extra error detection bit) can achieve a similar level of performance as the original EEC scheme with only one-fourth of the sketch size; on the other hand, with the same budget of transmission cost, the estimation accuracy of the original EEC scheme (288 bit design) can be greatly improved by our new estimators, and our gEEC design can achieve around 30% additional gain of estimation accuracy. Compared to the EToW's performance, the gEEC's performance is still better, especially in the range that θ is relatively large.

5.3 Implementation Cost of Estimators and Selection of Parameters

All presentations thus far have focused on the estimation performance. However, the implementation and the computation costs should also be considered. In practice, the MLE estimator can be implemented as table-based lookup. Whether or not it is enhanced by the Jeffreys prior, the MLE estimator can be transformed in this way:

$$\hat{\theta}_{MLE} = \arg\max_{\theta}\{\mathrm{A}(\theta)\mathrm{Y} + \mathrm{B}(\theta)\}, \tag{40}$$

where each entry of Y indicates the count of one particular type of sub-sketch equals to one particular value. $\mathrm{A}(\theta)$ and $\mathrm{B}(\theta)$ are determined by

We can implement (40) as a linear transform of $\mathbf{Y}$, i.e. $\mathbf{A}_{d_1\times d_2}\mathbf{Y}_{d_2\times 1} + \mathbf{B}_{d_1\times 1}$ and then find the maximum of the result. Here $\mathbf{A}_{d_1\times d_2}$ and $\mathbf{B}_{d_1\times 1}$ are both pre-calculated matrixes.

One of $\mathbf{A}$'s dimensions, d_1, is determined by the size of candidate θ's. For most practical EEC applications, d_1 in the order of

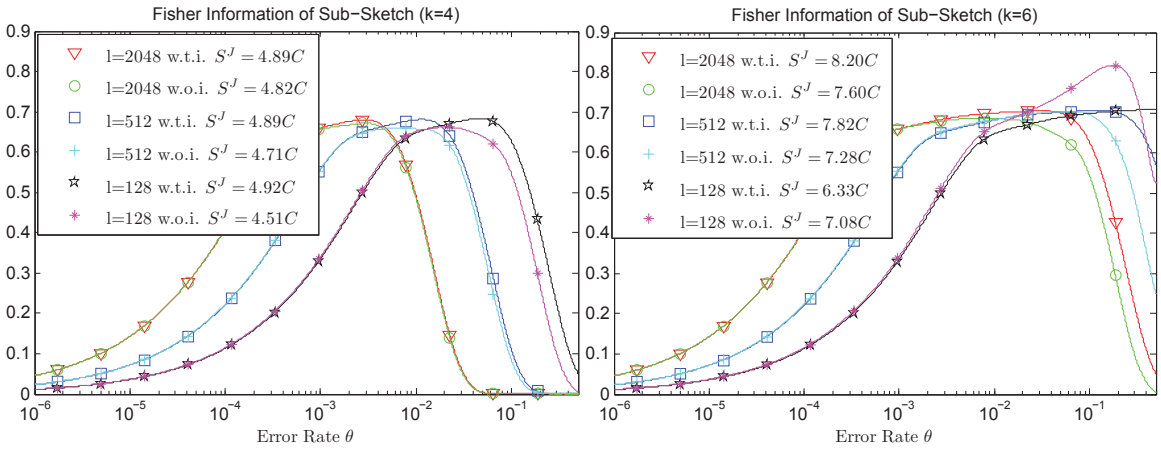

Figure 4: gEEC's Fisher information: Relationship to l and k in the cases with and without immunity (denoted as $w.t.i.$ and $w.o.i.$ respectively in the legend)

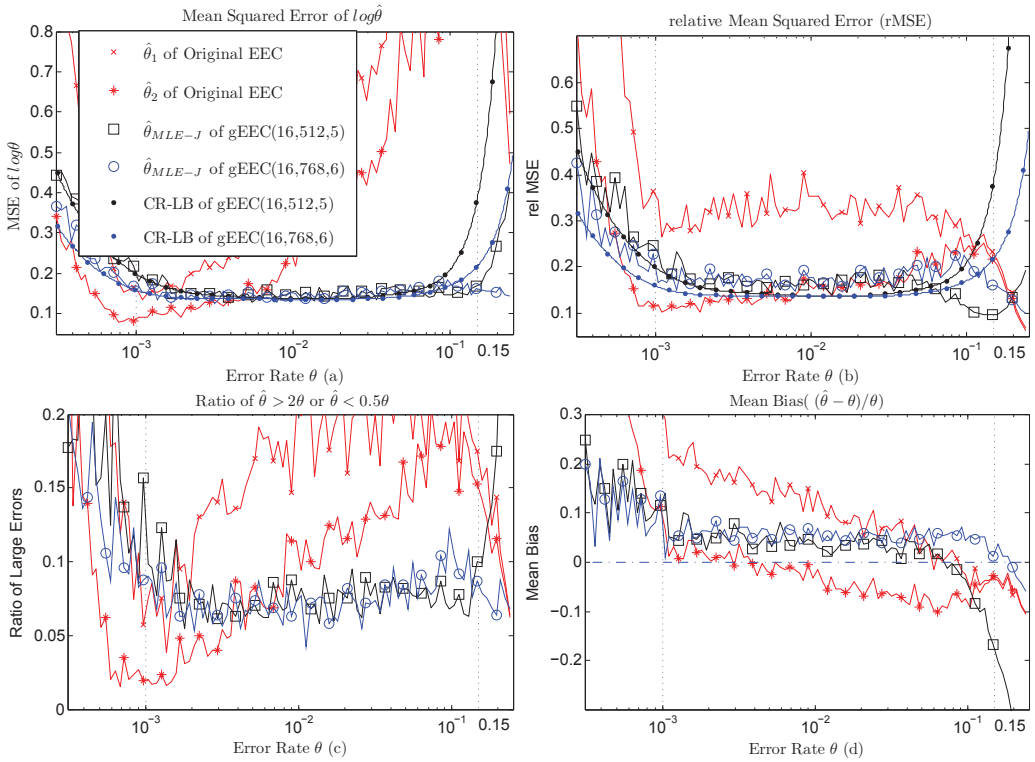

Figure 5: Empirical Results of Estimators: Compare the performance of original EEC (with original estimator) and two gEEC schemes (with much smaller size, 80 and 96 bits respectively), in the case without "immunity".

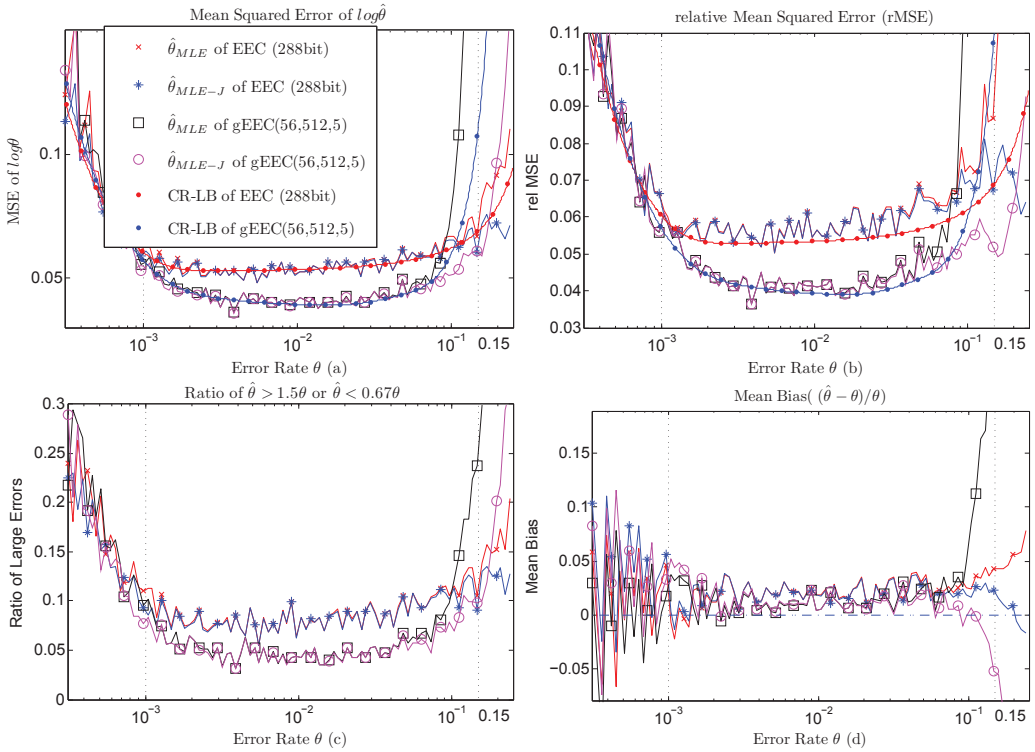

Figure 6: Empirical Results of Estimators: Compare the performance of two types of gEEC estimators (with or without Jeffrey's prior) on three schemes including the original EEC scheme, in the case without "immunity".

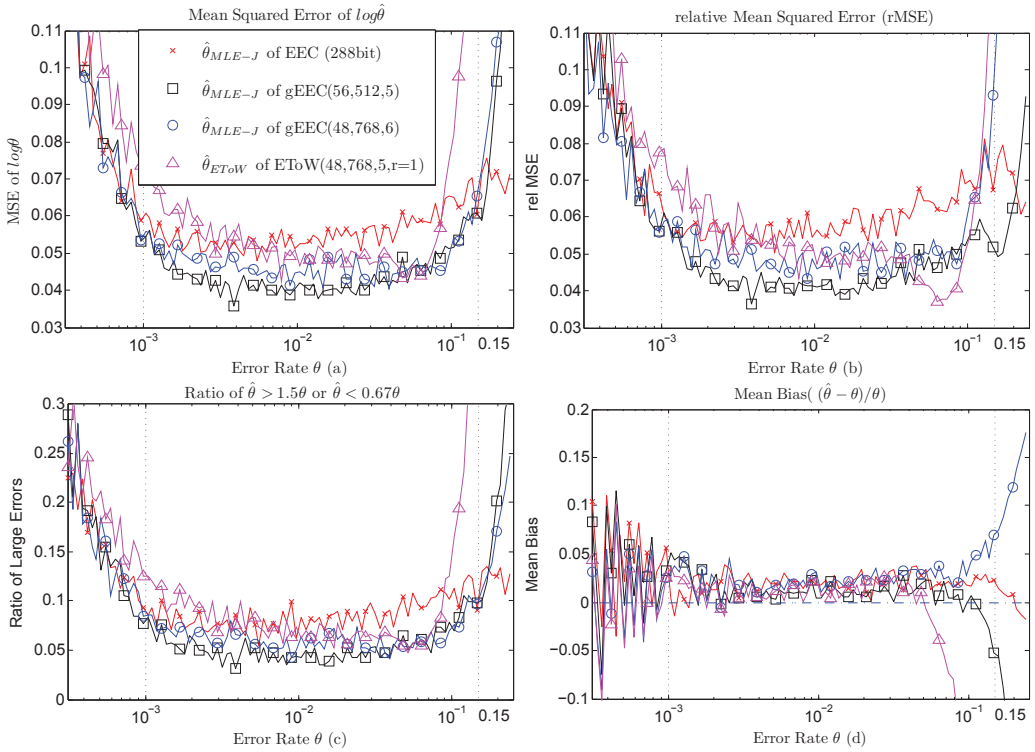

Figure 7: Empirical Results of Estimators: Compare the performance of four schemes (original EEC, gEEC(56,512,5),gEEC(48,768,6) and EToW), with almost the same size and using gEEC's estimator with Jeffrey's prior, in the case without "immunity".

hundreds is sufficient since it will be wasteful if it becomes more fine-grained than the spectrum of estimation.

$\mathbf{A}$'s other dimension, d_2, corresponding to the length of $\mathbf{Y}$, depends on the design of the sketch. Suppose the codeword is composed of c types of sub-sketches: m_1 gEEC(l_1, k_1) sub-sketches, m_2 gEEC(l_2, k_2) sub-sketches, $\ldots, m_c$ gEEC(l_c, k_c) sub-sketches. In the case of "immunity" where the estimator can directly infer from the difference between $\check{z}_i$ and z'_i, d_2 equals to $\sum_{i=1}^{c}(2^{k_i}-1)$. In the case without "immunity," d_2 equals to $\sum_{i=1}^{c}(2^{2k_i}-1)$, since in such a case the estimator should directly infer from the pair of $\check{z}_i$ and z'_i and hence the table size is squared.

On one hand, the MLE estimator above can be implemented with an extremely low cost, when all sub-sketches' k parameters equal 1, which means the scheme degenerates to the original EEC scheme. In this case, the matrix $\mathbf{A}$ can be as small as 9×100 (say, d_2=100). Moreover, some iterative methods similar to the bisection method can be employed to further reduce the number of multiplication, add and compare operations to about 90. Furthermore, since our estimator has strong built-in capability to combine the information from different levels, it can be shown that a 3-level, 96-bit-per-level design performs very closely to a 9-level 32-bit design, which can reduce the number of the sub-sketch types and hence even further reduce the cost by two-thirds.

On the other hand, as shown by previous evaluation results, larger values of k in the sub-sketch, such as 5 or 6, can generally bring around 25% of additional improvement of estimation accuracy, at the cost of thousands times higher cost in the storage and the lookups in tables $\mathbf{A}$ and $\mathbf{B}$. The computation cost actually remains almost the same since $\mathbf{X}$ is sparse, and we can implement the linear transform by summing up only few rows. Hence, the applicability of this improvement depends on the application scenario, since a lookup table of several hundreds KB might be a very small cost for some applications, but infeasible for others. Also, note that as discussed in Sec. 5.1, a large k might not be necessary if the target range of parameters is not so wide.

There are two middle paths between the cases above. One way is to use the combinations of several sub-sketches with $k = 3$ or 4 and different l's, the cost of which is much smaller than $k = 5$ since the table size increases exponentially ($O(2^{2k})$) in the case without immunity, while the scheme can still receive some gain on estimation accuracy. Another way is to use the EToW's scheme and estimator proposed in [6], whose implementation cost is much lower, though performance is also weaker, especially when the error rate θ is large.

In summary, the gEEC framework can be easily and flexibly configured for different requirements of estimation accuracy. All guidelines discussed in this section not only hold for θ in the range $[0.001, 0.15]$, but also for the other arbitrary ranges as needed.

6. CONCLUSION

The seminal work of Chen *et al.* [3] has opened the door for the design of high-quality error estimating codes, with applications towards improving wireless network performance. Chen *et al.* [3] designed an exceedingly simple code for estimating bit error rates in packets being transmitted, and it is an open—yet challenging—question whether this code is optimal in practice.

In this paper, we have systematically investigated the design space of error estimating codes, stemming from the natural question whether EEC achieves the best tradeoff between the space and estimation accuracy in estimating bit error rates. Along the path of our exploration using Fisher information analysis, we have demonstrated that EEC decoding is inefficient, and proposed a new estimator (decoder) that achieves a significantly higher accuracy. While investigating whether EEC encoding is efficient, we have developed a generalized coding framework, called generalized EEC, in which existing designs, such as EEC and EToW, are just degenerate cases. Using this unified framework, we found that some parameterization of gEEC similar to EToW can contain around 25% more information than the pure EEC scheme. To our knowledge, our work represents the first systematic study of the non-asymptotic optimality of error estimating codes.

Acknowledgement: The work of Hua and Xu is supported in part by the US National Science Foundation through grants CNS 0905169 and CNS 0910592.

7. REFERENCES

[1] ALON, N., GIBBONS, P. B., MATIAS, Y., AND SZEGEDY, M. Tracking join and self-join sizes in limited storage. *J. Comput. Syst. Sci. 64*, 3 (2002), 719–747.

[2] CHEN, B., ZHOU, Z., ZHAO, Y., AND YU, H. Technical report: Efficient error estimating coding: feasibility and applications.

[3] CHEN, B., ZHOU, Z., ZHAO, Y., AND YU, H. Efficient error estimating coding: feasibility and applications. In *SIGCOMM* (2010), pp. 3–14.

[4] COVER, T. M., AND THOMAS, J. A. *Elements of information theory (2. ed.)*. Wiley, 2006.

[5] FIRTH, D. Bias reduction of maximum likelihood estimates. *Biometrika 80*, 1 (1993), 27–38.

[6] HUA, N., LALL, A., LI, B., AND XU, J. A simpler and better design of error estimating coding. In *IEEE INFOCOM* (2012).

[7] KUSHILEVITZ, E., AND NISAN, N. *Communication complexity*. Cambridge University Press, 1997.

[8] LEHMANN, E., AND CASELLA, G. *Theory of Point Estimation, 2nd edition*. Springer, 1998.

[9] RIBEIRO, B. F., TOWSLEY, D. F., YE, T., AND BOLOT, J. Fisher information of sampled packets: an application to flow size estimation. In *Internet Measurement Conference* (2006), pp. 15–26.

[10] TUNE, P., AND VEITCH, D. Towards optimal sampling for flow size estimation. In *Internet Measurement Comference* (2008), pp. 243–256.

[11] TUNE, P., AND VEITCH, D. Sampling vs sketching: An information theoretic comparison. In *INFOCOM* (2011), pp. 2105–2113.