[3] A. El Gamal and M. Aref, "The capacity of the semi-deterministic relay channel," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 3, p. 536, May 1982.
[4] Y. Gabbai and S. I. Bross, "Achievable rates for the discrete memoryless relay channel with partial feedback configurations," *IEEE Trans. Inf. Theory*, submitted for publication.
[5] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 1, pp. 1–11, Jan. 1976.

A Counterexample to Cover's $2P$ Conjecture on Gaussian Feedback Capacity

Young-Han Kim, *Student Member, IEEE*

***Abstract*—We provide a counterexample to Cover's conjecture that the feedback capacity $C_{\rm FB}$ of an additive Gaussian noise channel under power constraint P be no greater than the nonfeedback capacity C of the same channel under power constraint $2P$, i.e., $C_{\rm FB}(P) \le C(2P)$.**

***Index Terms*—Additive Gaussian noise channels, channel capacity, conjecture, counterexample, feedback.**

I. Background

Consider the additive Gaussian noise channel

$$Y_i = X_i + Z_i, \quad i = 1, 2, \ldots$$

where the additive Gaussian noise process $\{Z_i\}_{i=1}^{\infty}$ is stationary. It is well known that feedback does not increase the capacity by much. For example, the following relationships hold between the nonfeedback capacity $C(P)$ and the feedback capacity $C_{\rm FB}(P)$ under the average power constraint P:

$$C(P) \le C_{\rm FB}(P) \le 2C(P)$$
$$C(P) \le C_{\rm FB}(P) \le C(P) + \frac{1}{2}.$$

(See Cover and Pombra [5] for rigorous definitions of feedback and nonfeedback capacities and proofs for the above upper bounds. Throughout this paper, the capacity is in bits and the logarithm is to base 2.)

These upper bounds on feedback capacity were later refined by Chen and Yanagi [3] as

$$C_{\rm FB}(P) \le \left(1 + \frac{1}{\alpha}\right) C(\alpha P)$$
$$C_{\rm FB}(P) \le C(\alpha P) + \frac{1}{2}\log\left(1 + \frac{1}{\alpha}\right)$$

Manuscript received November 4, 2005; revised March 29, 2006. This work was supported in part by the NSF under Grant CCR-0311633.
The author is with the Information Systems Laboratory, Stanford University, Stanford, CA 94305 USA (e-mail: yhk@stanford.edu).
Communicated by Y. Steinberg, Associate Editor for Shannon Theory.
Digital Object Identifier 10.1109/TIT.2006.878167

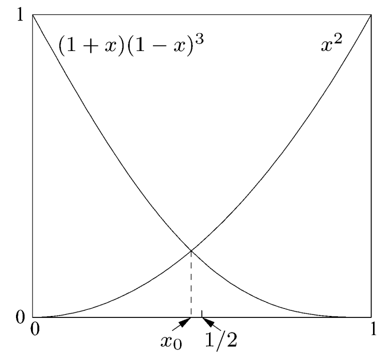

Fig. 1. Plot of x^2 vs. $(1+x)(1-x)^3$.

for any $\alpha > 0$. In particular, taking $\alpha = 2$, we get

$$C_{\rm FB}(P) \le \min\left\{\left(\frac{3}{2}\right) C(2P), C(2P) + \frac{1}{2}\log\left(\frac{3}{2}\right)\right\}.$$

In fact, Cover [4] conjectured that

$$C_{\rm FB}(P) \le C(2P)$$

and it has been long believed that this conjecture is true. (See Chen and Yanagi [8], [2] for a partial confirmation of Cover's conjecture.)

II. A Counterexample

Consider the stationary Gaussian noise process $\{Z_i\}_{i=1}^{\infty}$ with power spectral density

$$S_Z(e^{i\theta}) = |1 + e^{i\theta}|^2 = 2(1 + \cos\theta).$$

Now under the power constraint $P = 2$, it can be easily shown [1, Sec. 7.4] that the nonfeedback capacity is achieved by the water-filling input spectrum $S_X(e^{i\theta}) = 2(1 - \cos\theta)$, which yields the output spectrum $S_Y(e^{i\theta}) \equiv 4$ and the capacity

$$C(2) = \int_{-\pi}^{\pi} \frac{1}{2}\log\left(\frac{S_Y(e^{i\theta})}{S_Z(e^{i\theta})}\right)\frac{d\theta}{2\pi} = 1 \text{ bit}.$$

On the other hand, it can be shown [9] that the celebrated Schalkwijk–Kailath feedback coding scheme [6], [7] achieves the data rate

$$R_{\rm SK}(P) = -\log x_0$$

under power constraint P, where x_0 is the unique positive root of the equation

$$Px^2 = (1 + x)(1 - x)^3. \tag{1}$$

Now for $P = 1$, we can readily check that the unique positive root x_0 of (1) is less than $1/2$, since $f(x) := x^2 - (1 + x)(1 - x)^3$ is strictly increasing and continuous on [0, 1] with $f(0) = -1$ and $f(1/2) = 1/16$; see Fig. 1. Therefore,

$$C_{\rm FB}(1) \ge R_{\rm SK}(1) = -\log x_0 > 1 = C(2).$$

ACKNOWLEDGMENT

The author is grateful to Tom Cover for stimulating discussions. He also wishes to thank Prof. Kenjiro Yanagi and the Associate Editor Yossi Steinberg for their helpful comments.

REFERENCES

[1] R. E. Blahut, *Principles and Practice of Information Theory*. Reading, MA: Addison-Wesley, 1987.
[2] H. W. Chen and K. Yanagi, "On Cover's conjecture on capacity of Gaussian channel with feedback," *IEICE Trans. Fund.*, vol. E80-A, no. 11, pp. 2272–2275, 1997.
[3] ——, "Refinements of the half-bit and factor-of-two bounds for capacity in Gaussian channel with feedback," *IEEE Trans. Inf. Theory*, vol. IT-45, no. 1, pp. 319–325, Jan. 1999.
[4] T. M. Cover, "Conjecture: Feedback doesn't help much," in *Open Problems in Communication and Computation*, T. M. Cover and B. Gopinath, Eds. New York: Springer-Verlag, 1987, pp. 70–71.
[5] T. M. Cover and S. Pombra, "Gaussian feedback capacity," *IEEE Trans. Inf. Theory*, vol. IT-35, no. 1, pp. 37–43, Jan. 1989.
[6] J. P. M. Schalkwijk and T. Kailath, "A coding scheme for additive noise channels with feedback—I: No bandwidth constraint," *IEEE Trans. Inf. Theory*, vol. IT-12, pp. 172–182, Apr. 1966.
[7] J. P. M. Schalkwijk, "A coding scheme for additive noise channels with feedback—II: Band-limited signals," *IEEE Trans. Inf. Theory*, vol. IT-12, pp. 183–189, Apr. 1966.
[8] K. Yanagi, "On the Cover's conjecture of Gaussian feedback capacity," in *Proc. IEEE Int. Symp. Inf. Theory*, Ulm, Germany, 1997, p. 130.
[9] S. Yang, A. Kavcic, and S. Tatikonda, "Linear Gaussian channels: Feedback capacity under power constraints," in *Proc. IEEE Int. Symp. Inf. Theory*, Chicago, IL, June/July 2004, p. 72.

A Comparison of Methods for Redundancy Reduction in Recurrence Time Coding

Hidetoshi Yokoo, *Associate Member, IEEE*

***Abstract*—Recurrence time of a symbol in a string is defined as the number of symbols that have appeared since the last previous occurrence of the same symbol. It is one of the most fundamental quantities that can be used in universal source coding. If we count only the minimum required number of symbols occurring in the recurrence period, we can reduce some redundancy contained in recurrence time coding. The move-to-front (MTF) scheme is a typical example that shares the idea. In this correspondence, we establish three such schemes, and make a basic comparison with one another from the viewpoint that they can be thought of as different attempts to realize the above idea.**

***Index Terms*—Data compression, move-to-front (MTF), recency rank, recurrence time, source coding, universal codes.**

I. INTRODUCTION

Recurrence time coding, or interval coding proposed by P. Elias [7] is a method for universal lossless compression, which encodes recurrence times of symbols in a string. The idea is extended from symbols to strings in [12], where its asymptotic optimality with respect to the string length is shown. The well-known Lempel–Ziv code [14] is another type of realization of recurrence time coding [13].

Manuscript received March 11, 2005; revised April 16, 2006.
The author is with the Department of Computer Science, Gunma University, Kiryu 376-8515, Japan (e-mail: yokoo@cs.gunma-u.ac.jp).
Communicated by S. Savari, Associate Editor for Source Coding.
Digital Object Identifier 10.1109/TIT.2006.878177

It is known that recurrence time coding is redundant, although a kind of redundancy contained in symbolwise recurrence time coding is easy to reduce. The move-to-front (MTF) scheme, which was developed independently of recurrence time coding under the name of *book stack* [10] and has been known fairly formerly as various names in various contexts [8], can be regarded as an improvement of recurrence time coding. The recurrence time of a symbol is defined as the number of symbols that have appeared since the last previous occurrence of the same symbol. If we encode the number of distinct symbols occurring in that interval, we obtain the MTF scheme [3], [7]. Another improvement of recurrence time coding, which counts the number of alphabetically bigger or smaller symbols in the interval, is also known [2], [9].

These improvements of recurrence time coding share an idea that one should encode the minimum number of symbols required to specify the current symbol. Since the minimum required number naturally depends on how to specify symbols, redundancy reduction in recurrence time coding should be analyzed in relation to how to specify the symbols in a string.

In this correspondence, we begin with three different representations for specifying symbols via recurrence times, and establish their respective schemes for reducing redundancy, which include the MTF scheme. The three schemes have already been known, and actually applied to the block-sorting compression algorithm [4] as its second step component [2], [6]. However, they have not been analyzed and compared from the viewpoint that they can be thought of as different attempts to realize redundancy reduction in recurrence time coding. Especially, very little attention has been paid to theoretical aspects of the newer two schemes, whereas even a redundancy analysis with no symbol extension is given to the MTF scheme [1]. This correspondence explores these three improved schemes by highlighting their common natures, and compares relatively the expectations of their outputs for memoryless sources. We also reveal the entropies assumed by the three schemes on binary memoryless sources.

II. REPRESENTATIONS OF RECURRENCE TIME CODING

When we consider redundancy reduction in recurrence time coding, the representation of a recurrence time itself plays an essential role. In this section, we give three different representations of a recurrence time, which will be used to derive three different methods for redundancy reduction.

Let $\mathcal{A} = \{a_1, a_2, \ldots, a_\alpha\}$ denote a source alphabet of finite size α. Elements of the alphabet are called *symbols*. We assume that the symbols are totally ordered by some ordering relation. We say that the symbol a_i is *alphabetically smaller* than the symbol a_m for $1 \leq i < m \leq \alpha$, and conversely the symbol a_m is *alphabetically bigger* than the symbol a_i. The ordering relation is usually inherent in the alphabet. However, as we will see later, it may be determined after the data string is processed. For example, we can use actual symbol frequencies to define the order of symbols.

Suppose that we are going to encode an n-tuple of symbols (i.e., a *string*)

$$x_1^n = x_1 x_2 \cdots x_n, \quad x_k \in \mathcal{A},\ k = 1, 2, \ldots, n$$

on which we measure how far two positions are using the distance

$$d(k, j) = k - j, \quad 1 \leq j < k \leq n.$$

For the kth symbol $x_k = a \in \mathcal{A}$, if we have no a in $\{x_1, x_2, \ldots, x_{k-1}\}$ then we say that the symbol x_k is the *initial*