This is a preprint of a paper presented at the NATO Advanced Study Institute on New Directions in Signal Processing in Communications and Control, Darlington, England, August 5-17, 1974, that will appear in the Institute Proceedings to be published by the Noordhoff International Publishing Company of Holland.

THE CODEWORD AND SYNDROME METHODS FOR DATA COMPRESSION WITH ERROR-CORRECTING CODES*

James L. Massey

Freimann Professor of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana, U.S.A.

ABSTRACT

Error-correcting codes can be used in data compression systems either (1) by treating the data sequence as a received word, the codeword-method, or (2) by treating the data sequence as an error-pattern, the syndrome-method. The former is the traditional approach while the latter is a recently proposed alternative. Sufficient background in data compression and in error-correcting codes is given to enable a comparison to be made between these approaches. It is concluded from this comparison that the syndrome-method appears to be the more attractive for use with real data sources which tend to be highly asymmetric and to exhibit considerable memory effects.

LIST OF SYMBOLS

H	parity-check matrix of a linear code
p	reconstruction error probability for a data compression system
P_e	decoding error probability for an error-correcting code
$\underline{s}$	syndrome for an error-correcting code
V	a linear error-correcting code
γ	nominal compression ratio of a data compression system

* This research was supported by the U.S. National Aeronautics and Space Administration under NASA Grant NGL 15-004-026 at the University of Notre Dame in liaison with the Communications and Navigation Division of the Goddard Space Flight Center.

This is a preprint of a paper presented at the NATO Advanced Study Institute on New Directions in Signal Processing in Communications and Control, Darlington, England, August 5-17, 1974, that will appear in the Institute Proceedings to be published by the Noordhoff International Publishing Company of Holland.

1. INTRODUCTION

Our aim in this tutorial lecture is to explain how linear "error-correcting codes" may be used to accomplish "data compression" for information sources. In this section, we shall review some aspects of the data-compression problem. In the next section we review that part of the theory of linear error-correcting codes which will be needed in the sequel. In section 3, we describe the standard way that linear error-correcting codes have been proposed for use in data-compression, viz. the "codeword-method," and we discuss the advantages and limitations of this approach. In section 4, we describe a recently proposed way to use linear error-correcting codes for data-compression, viz. the "syndrome-method," and we argue that this approach appears more promising for use with real data sources.

Figure 1 illustrates the general data-compression situation. For simplicity we shall assume here and hereafter that the data source is a binary source, i.e. that each source output letter

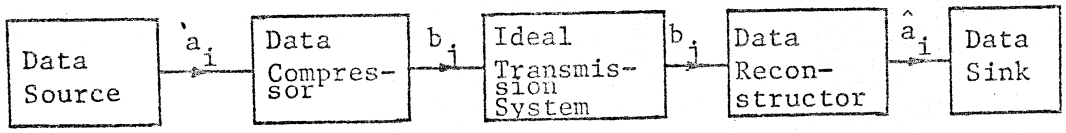

Fig. 1 The general data-compression situation

a_i is a binary digit. The data compressor maps the source output sequence into a second binary sequence $b_1, b_2, b_3, \ldots$ which we assume is transmitted perfectly to some remote location. At this location, the data reconstructor maps the "compressed data" $b_1, b_2, b_3, \ldots$ into its estimate $\hat{a}_1, \hat{a}_2, \hat{a}_3, \ldots$ of the actual source data sequence $a_1, a_2, a_3, \ldots$. This estimate can be faulty if the data compressor should map two or more source data sequences into the same compressed data sequence.

For a data compression system as in Figure 1, we can define its nominal-compression-ratio to be

$$\gamma = \lim_{n \to \infty} \frac{n}{k_n} \tag{1}$$

where k_n is the number of compressed digits resulting from n source digits. One wants of course for γ to be large, but this is not the only consideration. One could in fact make $\gamma = \infty$ by using

the trivial system such that $k_n = 0$ for all n; but of course the reconstruction of the source would then be very bad indeed! The nominal-compression-ratio, γ, is a true measure of the quality of the compression scheme only when the scheme always gives perfect reproduction of the source. When the source reconstruction can be imperfect, we must "correct" γ to reflect this fact. We do this by considering the average probability of an erroneously reconstructed digit

$$p = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \Pr(\hat{a}_i \neq \hat{a}_i). \tag{2}$$

According to information theory, cf. Gallager[1], the average number of binary digits to indicate whether a given digit a_i will be incorrectly reconstructed is

$$h(p) = -p \log p - (1-p) \log (1-p) \tag{3}$$

where here and hereafter the logarithms are taken to the base 2. The original data compression scheme can thus be converted to a scheme which always gives perfect reconstruction by adding nh(p) binary digits to the k_n digits in the compressed scheme resulting from n source digits when n is large. For this modified system, the nominal-compression-ratio is

$$\gamma_c = \frac{\gamma}{1 + \gamma\, h(p)} \tag{4}$$

where γ is the nominal-compression-ratio of the original scheme. We call γ_c the <u>corrected-compression-ratio</u> for the original data compression scheme, and we shall use γ_c as our measure of the quality of a data compression scheme. Note that $\gamma_c = \gamma$ when $p = 0$, i.e. when the reconstruction is always perfect. The use of γ_c allows us to answer such questions as: Which is better, a scheme with $\gamma = 5$ and $p = 10^{-2}$ or a scheme with $\gamma = 4$ and $p = 0$? For the former scheme, we find $1 + \gamma\, h(p) = 1.404$ so that $\gamma_c = 3.56$ which is considerably inferior to the $\gamma_c = \gamma = 4$ of the latter system.

2. LINEAR ERROR-CORRECTING CODES

Figure 2 illustrates the general situation for the use of binary error-correcting codes in one-way communications or "forward error correction." The encoder maps the source sequence $u_1, u_2, u_3, \ldots$ into a second binary sequence, the encoded sequence,

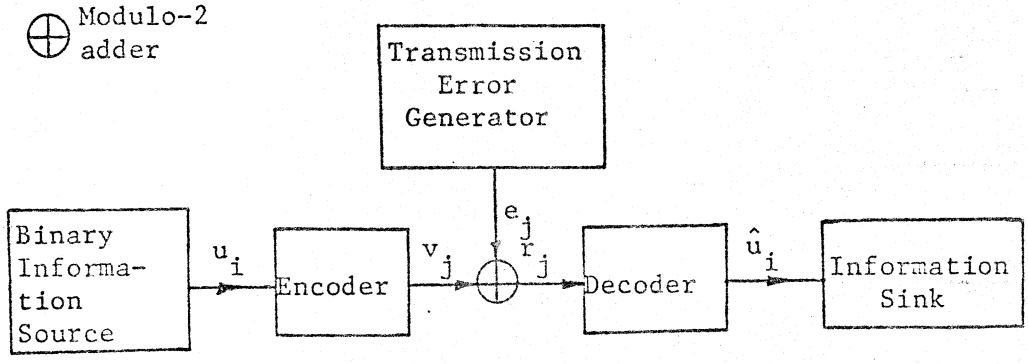

Fig. 2 General binary error-correcting system

$v_1, v_2, v_3, \ldots$. The Transmission error sequence $e_1, e_2, e_3, \ldots$ represents the effect of the channel on the encoded sequence in that the digits in the received sequence $r_1, r_2, r_3, \ldots$ satisfy $r_j = v_j$ when $e_j = 0$ and $r_j \neq v_j$ when $e_j = 1$. The <u>rate</u> of the coding system is defined to be

$$R = \lim_{K \to \infty} \frac{K}{N_K} \tag{5}$$

where N_K is the number of encoded digits that result from K source digits. Let $\hat{v}_1, \hat{v}_2, \hat{v}_3, \ldots$ be the encoded sequence that would result from encoding the decoded source sequence $\hat{u}_1, \hat{u}_2, \hat{u}_3, \ldots$. We define the average decoding error probability to be

$$P_e = \lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} \Pr(\hat{v}_i \neq v_i). \tag{6}$$

A block coding system is an error-correcting system in which each span of K digits from the source is independently mapped by the encoder into a span or "block" of N encoded digits. The rate of such a system is

$$R = \frac{K}{N}. \tag{7}$$

The <u>block code</u> is defined to be the set of 2^K "codewords" $\underline{v} = (v_1, v_2, \ldots v_N)$ resulting from all possible choices of $\underline{u} = (u_1, u_2, \ldots u_K)$. Note that many different encoders will give the same code.

A linear block code, or (N,K) code, is a block code for which the code, V, is a K-dimensional vector space over the binary number field, GF(2). If H is any (N-K) x N binary matrix such that V is the kernel of H, i.e. such that V is the set of all $\underline{v}$ for which $H\underline{v} = \underline{0}$, then H is called a (reduced) parity-check matrix for the linear code V. If G is any N x K binary matrix whose columns are a basis for V, then G is called an encoding matrix, or "generator matrix," for the linear code V; this terminology stems from the fact that $\underline{v} = G\underline{u}$ describes a "linear encoding rule" for the code V.

Let $\underline{e} = (e_1, e_2, \ldots e_N)$ be the error pattern affecting the codeword $\underline{v}$. Then the received word $\underline{r} = (r_1, r_2, \ldots r_N)$ is

$$\underline{r} = \underline{v} \oplus \underline{e} \tag{8}$$

where the addition is term-by term addition modulo 2. We say that the code V can correct a set of error patterns $\varepsilon = \{\underline{e}_1, \underline{e}_2, \ldots \underline{e}_m\}$ if there exists a decoder such that $\hat{\underline{v}} = \underline{v}$ whenever $\underline{e}$ is some error pattern in ε. The coset containing the error pattern $\underline{e}_i$ is

$$C(\underline{e}_i) = \underline{e}_i \oplus V \tag{9}$$

where the notation on the right means the set of all 2^K sums $\underline{e}_i \oplus \underline{v}$ with $\underline{v} \in V$. Note that $C(\underline{e}_i)$ is just the set of all received words $\underline{r}$ that can result when $\underline{e} = \underline{e}_i$. It follows that the code V can correct all errors in ε if and only if the m error patterns in ε are contained in disjoint cosets. Note that it is always the case either that $C(\underline{e}_i) = C(\underline{e}_j)$ or that $C(\underline{e}_i) \cap C(\underline{e}_j) = \emptyset$; that is, cosets either coincide or are disjoint because if there are $\underline{v}_1$ and $\underline{v}_2$ in V such that $\underline{e}_i \oplus \underline{v}_1 = \underline{e}_j \oplus \underline{v}_2$ then

$$\underline{e}_i \oplus V = \underline{e}_j \oplus \underline{v}_2 \oplus \underline{v}_1 \oplus V = \underline{e}_j \oplus V.$$

Consider now a selected parity-check matrix H for V. We define the syndrome for the received word $\underline{r}$ to be

$$\underline{s} = H\underline{r}. \tag{10}$$

It then follows that

$$\begin{aligned} \underline{s} &= H(\underline{v} \oplus \underline{e}) \\ &= H\,\underline{v} \oplus H\,\underline{e} \\ &= H\,\underline{e}, \end{aligned} \tag{11}$$

where we have used the fact that $H\,\underline{v} = \underline{0}$, which shows that the syndrome depends only on the error pattern. It follows now from (9) that all error patterns in the same coset have the same syndrome. Conversely, if two error patterns have the same syndrome then they lie in the same coset for suppose $H\,\underline{e}_1 = H\,\underline{e}_2$, then $H(\underline{e}_1 \oplus \underline{e}_2) = \underline{0}$ which implies $\underline{e}_1 \oplus \underline{e}_2 = \underline{v}$ for some $\underline{v}$ in V which in turn implies $\underline{e}_1 = \underline{e}_2 + \underline{v}$ which verifies that $\underline{e}_1$ and $\underline{e}_2$ both lie in $C(\underline{e}_2)$. It follows then that <u>the code V can correct the set of error patterns ε if and only if the error patterns in ε have distinct syndromes</u>.

The preceding result is the foundation for a general decoding method for a linear code V which is known as <u>syndrome decoding</u> and which is illustrated in Figure 3. The received word is first processed with the parity-check matrix H to yield the syndrome $\underline{s}$.

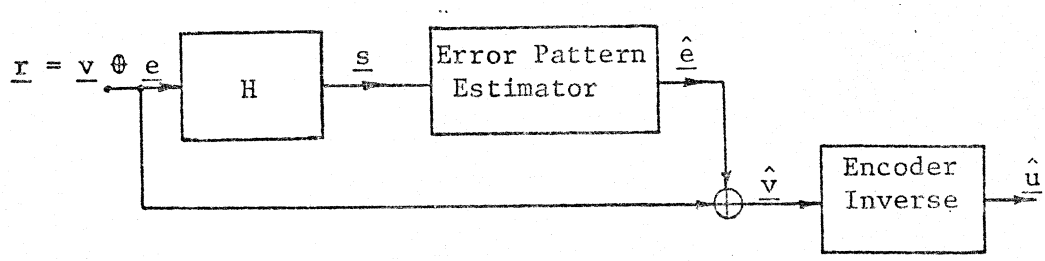

Fig. 3 A syndrome decoder for a block linear code V

Since $\underline{s}$ identifies the coset containing the error pattern $\underline{e}$, the <u>error pattern estimator</u> can emit as its estimate $\hat{\underline{e}}$ of $\underline{e}$ that element of ε which is in the coset $C(\underline{e})$. The estimate $\hat{\underline{v}} = \underline{r} \oplus \hat{\underline{e}} = \underline{v} \oplus \underline{e} \oplus \hat{\underline{e}}$ of $\underline{v}$ will then be correct if and only if $\underline{e} = \hat{\underline{e}}$. Finally, one passes $\hat{\underline{v}}$ through the inverse of the particular encoder being used for V to obtain the estimate $\hat{\underline{u}}$ of the information digits. It follows from the result of the previous paragraph that if there is any decoder for V which corrects all error patterns in a set ε there is also a syndrome decoder which corrects all error patterns in ε.

In general, the complexity of the error pattern estimator dominates the complexity of a syndrome decoder since, for any sensible decoder, this device implements a nonlinear function of its input that generally requires considerable logical circuitry. By contrast, the syndrome calculator (which implements the multiplication of $\underline{r}$ by H) and the encoder inverse are linear devices which are simply implemented. In particular, when the encoder is <u>systematic</u> in the sense that $\underline{u}$ forms the first K digits in $\underline{v}$, the encoder inverse is entirely trivial.

The minimum distance, d, of a block code is the minimum Hamming distance between pairs of its codewords, i.e. the fewest number of positions in which two codewords differ. If the code is a linear code V, d is equal to the minimum Hamming weight (i.e. minimum number of non-zero digits) of its codewords excluding $\underline{0}$. A block code can correct all patterns of t or fewer errors, i.e. correct the set ε of all error patterns of Hamming weight t or less, if and only if $2t < d$. Thus, a large minimum distance is important in error-correcting applications of the code.

3. THE CODEWORD-METHOD OF DATA COMPRESSION

The traditional, and the obvious way, to consider using a linear code V as the heart of a data compression scheme is (1) to treat the data source output sequence $(a_1, a_2, \ldots a_N)$ as the received word $\underline{r}$, (2) to use as the data compressor a decoder for V so that the compressed data $(b_1, b_2, \ldots b_K) = \hat{u}_K$, and (3) to use an encoder for V as the data reconstructor so that $(\hat{a}_1, \hat{a}_2, \ldots \hat{a}_K) = \hat{\underline{v}}$. We call this the codeword-method of using a linear code V for data compression purposes. From equations (1) and (7), we see that the nominal compression ratio is

$$\gamma = \frac{1}{R}. \tag{12}$$

The reconstruction error probability p can generally be found only by simulation. In Figure 4, we show the structure of a data compression system using the codeword-method approach.

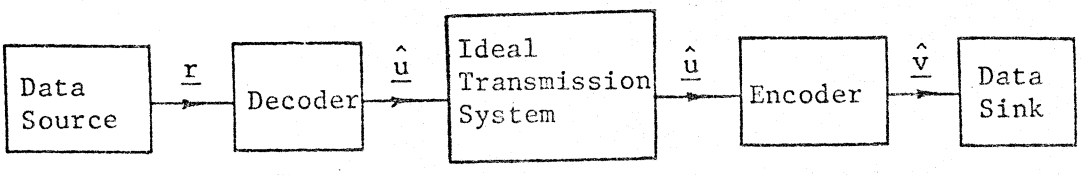

Fig. 4 The codeword-method of data compression

We see from Figure 4 that for the codeword-method to work effectively, there must be a codeword $\underline{v}$ near (in the sense of Hamming distance) to every likely source sequence $\underline{r}$. The code V will work effectively with some decoder if and only if it gives a good covering, in this sense, of the set of likely source sequences.

Recall that the code V works well for error-correction if it gives a good separation of its codewords as measured by d. In general, a code with good covering may have poor separation (and conversely.) For instance, if for any code V with minimum distance d, we add the vector $(1,0,0,\ldots 0)_{K+1}$ to a basis for V, then we obtain a linear code V' of rate $R' = \frac{K+1}{N} = R + \frac{1}{N}$ with minimum distance $d' = 1$. But V' covers the source at least as well as V since $V \subset V'$ and V' has almost the same nominal compression ratio as V so that V' is about as good as V for data compression purposes. Yet $d' = 1$ so that V' is useless for error-correcting purposes, not being even single-error-correcting. Thus, there is no reason to believe that linear codes V which are good for data compression purposes will be good for error-correcting purposes (and conversely.)

We also note, from Fig. 4, that the complex component (i.e. the decoder for V) is on the source side while the simple component (i.e. the encoder for V) is on the sink side in the codeword-method of data compression. This would be undesirable in space-telemetry applications (and similar situations) where one wants the data compressor aboard the spacecraft to be very simple whereas one might be willing to invest in considerable complexity for the data reconstructor at the ground station. On the other hand, in a "broadcasting" situation, it might be desirable to have very simple data reconstructors at the many data sink sites at the expense of having a complex data compressor at the source.

The Achilles' heel of the codeword-method of data compression, however, is that there seem to be few, if any, real data sources whose likely source sequences are well-covered by the codewords of a linear code with low-rate R (and hence high corrected compression ratio.) The first reason for this is that the vector space structure of V imposes considerable symmetry upon its set of codewords (e.g. in any position, half of the codewords contain a 0 and half contain a 1 unless all contain 0's in that position) whereas real sources tend to be highly asymmetric. Suppose a source, for instance, tends with high probability to emit 0's so that the sequence $\underline{0}$ and the N sequences with a single 1 are the most likely source sequences $\underline{r}$. For V to cover this source well, V must contain the N sequences with a single 1 as codewords and hence must be the vector space of dimension $K = N$ so that $R = 1$ and no compression is possible using V. The second reason is that real sources tend to exhibit considerable memory whereas the vector space structure of V tends to make its set of codewords resemble the likely output sequences of a source with little memory. For instance, suppose when the source emits a 1 it is likely to emit N or more consecutive 1's and such strings of 1's are separated by rather long strings of 0's. If V is to cover this source well, it must contain as codewords the N vectors (1,1,1,...1,1), (0,1,1, ...1,1), (0,0,1,...1,1), ... (0,0,0,...,0,1). Since these vectors

are again linearly independent, we must have $K = N$ and hence $\gamma = 1$ so that again no compression is possible.

One could of course consider the use of a nonlinear block code rather than a linear code in the codeword-method of data compression by taking the decoder and encoder in Fig. 4 to be those for the nonlinear code. In this case, it would be possible in principle to make the scheme effective (i.e. to achieve a corrected compression ratio nearly equal to the reciprocal of the source rate $H_\infty(A)$ in the terminology of our preceeding lecture) for any binary information source regardless of its assymmetry and memory properties. The "catch" however is that we know at present very little about nonlinear codes or about how to implement encoders and decoders for nonlinear codes that would not be prohibitively complex. The implementation of a decoder for a nonlinear code appears to be a far more complex task than implementing, say, a syndrome decoder for a linear code; and even the implementation of the encoder for a nonlinear code could in general be a difficult task in contrast with the inherent simplicity of linear encoders for linear codes.

4. THE SYNDROME-METHOD OF DATA COMPRESSION

Recently, Ohnsorge[2] and Ancheta[3] independently proposed a new way to use linear error-correcting codes for data compression. This method, which we shall call the syndrome-method, is (1) to treat the data source output sequence $(a_1, a_2, \ldots a_N)$ as the error pattern $\underline{e}$, (2) to use as the data compressor the syndrome-forming circuit of a syndrome decoder for the linear code, and (3) to use the error pattern estimator of the same syndrome decoder as the source reconstructor so that $(\hat{a}_1, \hat{a}_2, \ldots \hat{a}_N) = \hat{\underline{e}}$. In Figure 5, we show the structure of a data compression scheme using the syndrome-method approach.

Since the syndrome $\underline{s} = H\,\underline{e}$ for a linear (N,K) code is an N - K place vector, it follows from equation (1) and (7) that the nominal compression ratio is

$$\gamma = \frac{1}{1-R} \tag{13}$$

when the syndrome method is used. Moreover, from equations (2) and (6) and from the fact that $\hat{v}_i = v_i$ when and only when $\hat{e}_i = e_i$, it follows that the reconstruction error probability is

$$p = P_e \tag{14}$$

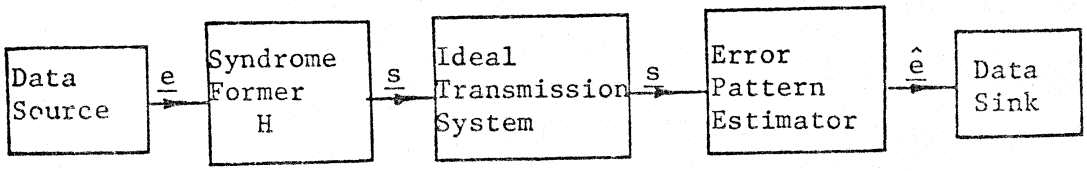

Fig. 5 The syndrome-method of data compression

where P_e is the decoding error probability for the syndrome decoder as in Figure 3 when the data source is used as the transmission error generator of Figure 2. Thus, the corrected compression ratio, according to (4), becomes

$$\gamma_c = \frac{1}{1 + h(P_e) - R} . \qquad (15)$$

Equation (15) demonstrates the desirability of using a high-rate linear code in the syndrome-method provided that the set ε of errors which this code can correct contains most of the high probability source output sequences.

We see that for the syndrome-method to work effectively the likely source output sequences must be well-covered by a set ε of error patterns correctable by the linear code V, in contrast to the codeword-method where this set of source sequences must be well-covered by V itself. We believe that the syndrome-method will be well-suited to the data compression of many real information sources since the sets ε of error patterns corrected by linear codes tend to be highly asymmetrical (in contrast to the symmetry of V itself) and to resemble sequences from sources with considerable memory. For instance, many linear codes are known for which ε can be taken as almost all of the sequences of Hamming weight t or less where $t << N$. Such codes would work well with asymmetrical sources for which an output of a 1 was much more likely than an output of 0. Again for instance, many linear codes are known for which ε is the set of almost all of the "bursts" of length b or less, i.e. where the non-zero digits of $\underline{e}$ are confined to at most b consecutive positions. Such codes would work well with "sporadic" sources which emit only 0's except for brief periods of activity.

It should also be noted that, in the syndrome-method of data compression, the simple component (i.e. the syndrome-former) is on the source side, whereas the complex component (i.e. the error pattern estimator) is on the sink side. We have already mentioned cases where this would be desirable and where this would be undesirable.

While it is yet too early to pronounce on the general effectiveness of the syndrome-method of data compression for real data sources, the results of Ancheta[3] give cause for some optimism. Ancheta considered the syndrome-method of data compression using a very simple Iwadare-Massey syndrome decoder (cf. Gallager[1]) for a sporadic source. He found that the true compression ratio was substantially superior to that obtained by run-length coding which is the usual data compression method used for such sources. It seems to us that the syndrome-method may afford a technique by which the vast quantity of known results for linear error-correcting codes may be put to good use in data compression applications.

5. REFERENCES

1. Gallager, R. G., Information Theory and Reliable Communication, John Wiley and Sons, New York, 1968.

2. Ohnsorge, H., A data compression system for the transmission of digitalized signals, Proceedings IEEE International Conference on Communications, 2, 48.5, 1973.

3. Ancheta, T. C., Jr., Syndrome-source-coding for data compression, Technical Report EE-7313 (Dept. of Electrical Engr., Univ. of Notre Dame, Notre Dame, Indiana), 1973.

6. BIBLIOGRAPHY

1. Berger, T., Rate Distortion Theory, Prentice-Hall, Englewood Cliffs, New Jersey, 1971.

2. Peterson, W. W. and Weldon, E. J., Jr., Error-Correcting Codes, M.I.T. Press, Cambridge, Massachusetts, 1972.