THE BELL SYSTEM TECHNICAL JOURNAL

MARCH 1981
VOL. 60, NO. 3

THE BELL SYSTEM TECHNICAL JOURNAL

THE BELL SYSTEM TECHNICAL JOURNAL is published monthly, except for the May-June and July-August combined issues, by the American Telephone and Telegraph Company, C. L. Brown, Chairman and Chief Executive Officer; W. M. Ellinghaus, President; V. A. Dwyer, Vice President and Treasurer; F. A. Hutson, Jr., Secretary. Editorial inquiries should be addressed to the Editor, The Bell System Technical Journal, Bell Laboratories, 600 Mountain Ave., Murray Hill, N.J. 07974. Checks for subscriptions should be made payable to The Bell System Technical Journal and should be addressed to Bell Laboratories, Circulation Group, Whippany Road, Whippany, N.J. 07981. Subscriptions $20.00 per year; single copies $2.00 each. Foreign postage $1.00 per year; 15 cents per copy. Printed in U.S.A. Second-class postage paid at New Providence, New Jersey 07974 and additional mailing offices.

ISSN0005-8580

Comments on the technical content of any article or brief are welcome. These and other editorial inquiries should be addressed to the Editor, The Bell System Technical Journal, Bell Laboratories, 600 Mountain Avenue, Murray Hill, N.J., 07974. Comments and inquiries, whether or not published, shall not be regarded as confidential or otherwise restricted in use and will become the property of the American Telephone and Telegraph Company. Comments selected for publication may be edited for brevity, subject to author approval.

THE BELL SYSTEM TECHNICAL JOURNAL

DEVOTED TO THE SCIENTIFIC AND ENGINEERING ASPECTS OF ELECTRICAL COMMUNICATION

Volume 60 March 1981 Number 3

On the Use of Dynamic Time Warping for Word Spotting and Connected Word Recognition*

By C. S. MYERS, L. R. RABINER, and A. E. ROSENBERG

(Manuscript received September 9, 1980)

Several variations on algorithms for dynamic time warping for speech processing applications have been proposed. This paper compares two of these algorithms, the fixed-range method and the local minimum method. We show that, based on results from some simple word spotting and connected word recognition experiments, the local minimum method performs considerably better than the fixed-range method. We describe explanations of this behavior and techniques for optimizing the parameters of the local minimum algorithm for both word spotting and connected word recognition.

I. INTRODUCTION

Time registration of a test and a reference pattern is one of the fundamental problems in the area of automatic speech recognition. This problem is important because the time scales of a test and a reference pattern are not perfectly aligned. In some cases the time scales can be registered by a simple linear compression or expansion[1,2]; however, in most cases, a nonlinear time warping is required to compensate for local compression or expansion of the time scale. For such cases, the class of algorithms known as dynamic time warping (DTW) methods has been developed. Work by Sakoe and Chiba,[3]

* The work presented here is based, in part, on the MS thesis, "A Comparative Study of Several Dynamic Time Warping Algorithms for Speech Recognition," by C. S. Myers, MIT, April 1980.

Itakura,[4] and White and Neely[2] has shown that DTW algorithms are an effective method of time registering patterns in isolated word recognition systems. Bridle[5] and Christiansen and Rushforth[6] have studied the applicability of DTW algorithms to word spotting, and recently, Sakoe,[7] Rabiner and Schmidt,[8] and Myers and Rabiner,[9] have successfully applied dynamic time-warping techniques to connected digit recognition. A great deal of work has been done in the area of performance evaluation of the various DTW algorithms as applied to discrete word recognition.[10-12] However, the effects of the DTW parameters on the overall performance of the algorithm for either word spotting or connected word recognition are not as well understood. The purpose of this paper is to discuss several proposed methods of applying DTW algorithms to word spotting and connected word recognition, and to study some of the factors which determine the performance of these algorithms.

The organization of this paper is as follows. In Section II we review the basic dynamic programming method of time alignment and show how it may be used efficiently in either a word spotting or a connected word recognition problem. We describe, in detail, two different DTW algorithms for which we have performed extensive evaluations. Section III contains a description of the experiments which we performed to evaluate the performance of the different DTW algorithms and the effects of the parameters associated with them. In Section IV we summarize the results of these experiments and draw some general conclusions on the use of DTW algorithms for word spotting and connected word recognition.

II. DYNAMIC PROGRAMMING FOR TIME ALIGNMENT

In this section we first review the basic principles of DTW algorithms as applied to discrete word recognition, and then point out some of the inherent difficulties involved in applying these algorithms to word spotting and connected speech recognition. We then show how it is possible to modify the basic DTW idea so that it may be used for both connected word recognition and word spotting applications.

2.1 Dynamic time warping for discrete word recognition

The problem of time alignment for discrete word recognition is illustrated in Fig. 1. A reference pattern, $\mathbf{R}(n)$, $n = 1, 2, \cdots, N$, consisting of a time sequence (i.e., frames) of a multidimensional feature vector is to be time registered with a test pattern, $\mathbf{T}(m)$, $m = 1, 2, \cdots, M$, which is also represented as a time sequence of a multidimensional feature vector. In Fig. 1, for the sake of clarity, both $\mathbf{R}(n)$ and $\mathbf{T}(m)$ are shown as one-dimensional functions. We shall assume that both the reference and the test pattern are measured from

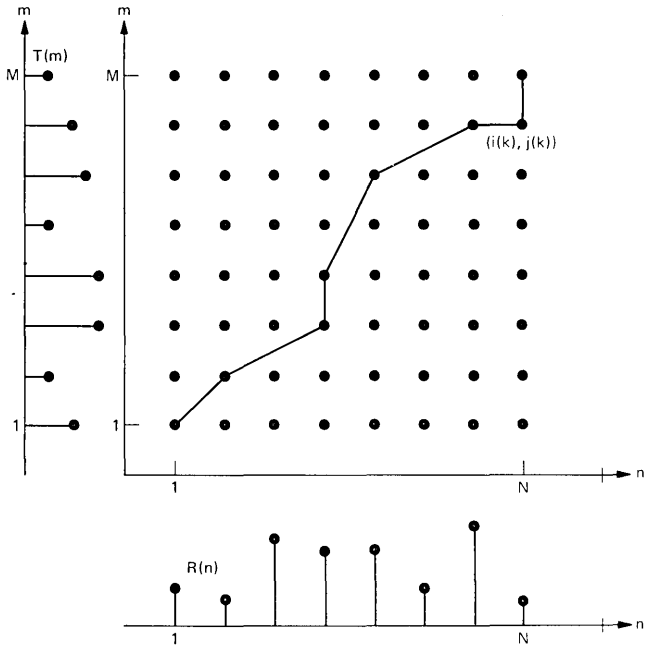


Fig. 1—Time warping of a reference and a test pattern.

the acoustic waveform of a single word, spoken in isolation, and that both the beginning and ending points of the reference and the test pattern have been accurately determined. The problem of time alignment is to find the path, here parameterized by the function pair $(i(k), j(k))$, which minimizes a given distance metric. A typical distance metric* is of the form

$$D(i(k), j(k)) = \frac{\sum_{k=1}^{K} d(i(k), j(k))\, \tilde{W}(k)}{\mathbf{N}(\tilde{W})}, \tag{1}$$

where K is the length of the path, $d(i(k), j(k))$ is the local distance, or dissimilarity, between frame $i(k)$ of the reference pattern and frame $j(k)$ of the test pattern, $\tilde{W}(k)$ is a weighting function applied to the path, and $\mathbf{N}(\tilde{W})$ is a normalization factor which is based on the particular weighting function that is chosen.

In addition to minimizing the global distance, the time alignment path is chosen to have certain desirable properties. One important property is the proper time registration of the beginning and ending points of the test and reference patterns, i.e.,

* D is shown here as a functional of the path function pair $(i(k), j(k))$.

$$i(1) = 1, \qquad j(1) = 1, \tag{2a}$$

$$i(K) = N, \qquad j(K) = M. \tag{2b}$$

Also, the time alignment path is required to obey certain shape and slope constraints. For example, it would not be reasonable to allow a path for which a 10 to 1 expansion or compression of the time axis occurs. Another consideration is the preservation of time order, i.e., the functions $i(k)$ and $j(k)$ must both be monotonically increasing.

These local continuity constraints are generally described by specifying the full path in terms of simple local paths which may be pieced together to form larger paths. For example, to reach a grid print (n, m) it may be reasonable to have come from any of the grid points $(n - 1, m - 1)$, $(n - 1, m - 2)$, or $(n - 2, m - 1)$, as shown in Fig. 2, part a. We refer to these constraints as Type I local constraints. Some other proposed sets of local constraints are shown in parts b, c, and d of Fig. 2. The crossed out arc in part d signifies the restriction that a path may not move horizontally for two consecutive segments.[4] All these local constraints limit the overall slope of the time alignment contour

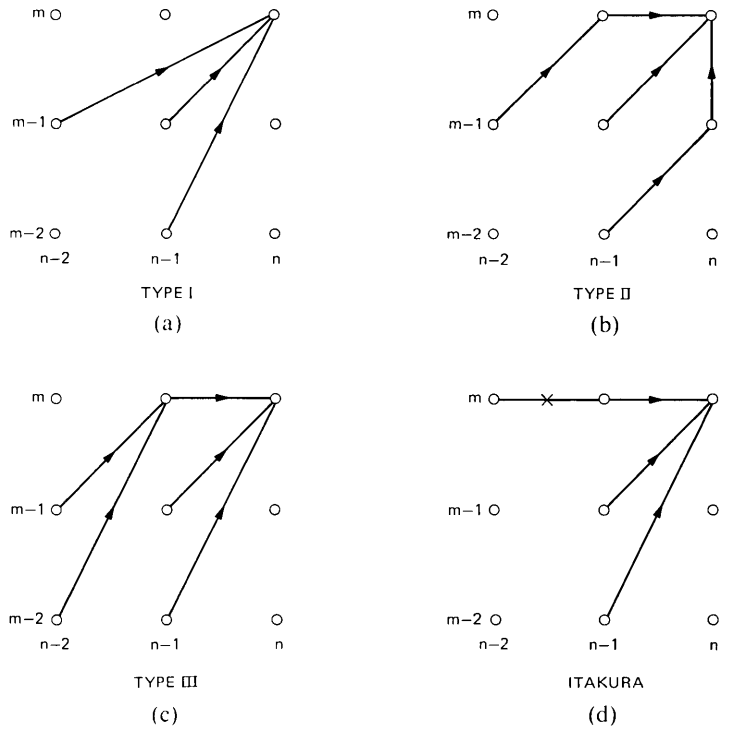


Fig. 2—Local constraints used for dynamic time warping.

to be between ½ and 2, in accordance with the results found by Sakoe and Chiba.[9]

To solve for the optimal time-alignment path, both the weighting function, $\tilde{W}(k)$, and the normalization factor, $\mathbf{N}(\tilde{W})$, must be specified in addition to the local constraints. Typically $\tilde{W}(k)$ is chosen to be either of two functions, i.e.,

$$\tilde{W}(k) = i(k) - i(k-1) \qquad \text{(Type a)}, \tag{3a}$$

$$\tilde{W}(k) = i(k) - i(k-1) + j(k) - j(k-1) \qquad \text{(Type b)}. \tag{3b}$$

These two weighting functions are referred to as the asymmetric weighting function, Type a, and the symmetric weighting function, Type b, and were originally proposed by Sakoe and Chiba.[3] Weighting function Type a weights all frames of the reference pattern equally, while weighting function Type b weights all frames of *both* the reference and the test equally. For initialization purposes, $i(0)$ and $j(0)$ are defined to be 0 and thus $\tilde{W}(1) = 1$ for weighting function Type a and $\tilde{W}(1) = 2$ for weighting function Type b.

The choice of $\mathbf{N}(\tilde{W})$ is typically made such that $D(i(k), j(k))$ is the average local distance along the path defined by $i(k)$ and $j(k)$, and is independent of both the lengths of the reference and test patterns, as well as the length of the time alignment path itself. The natural choice for $\mathbf{N}(\tilde{W})$ is thus

$$\mathbf{N}(\tilde{W}) = \sum_{k=1}^{K} \tilde{W}(k). \tag{4}$$

For weighting functions Types a and b the normalization is given by

$$\mathbf{N}(\tilde{W}_a) = \sum_{k=1}^{K} (i(k) - i(k-1)) = i(K) - i(0) = N, \tag{5a}$$

$$\begin{aligned}\mathbf{N}(\tilde{W}_b) &= \sum_{k=1}^{K} (i(k) - i(k-1) + j(k) - j(k-1)) \\ &= i(K) - i(0) + j(K) - j(0) = N + M.\end{aligned} \tag{5b}$$

Given a weighting function and a set of local constraints it is possible to define the optimal time-alignment path as that path which minimizes the total distance $D(i(k), j(k))$. More formally, if we denote the distance associated with the optimal path as $\hat{D}$, then

$$\hat{D} = \min_{K, i(k), j(k)} [D(i(k), j(k))]. \tag{6}$$

The solution to this problem may be found by dynamic programming by use of the following optimality principle:

Local Optimality: If the best path from the grid point (1, 1) to the grid point (n, m) goes through a grid point (n', m'), then the best path

from the grid point (1, 1) to the grid point (n, m) includes, as a portion of it, the best path from the grid point (1, 1) to the grid point (n', m').

Thus, if we define $D_A(n, m)$ as the minimum total distance along any path from the grid point (1, 1) to the grid point (n, m), then $D_A(n, m)$ can be computed, recursively according to the optimality principle, as

$$D_A(n, m) = \min_{n',m'} [D_A(n', m') + \hat{d}((n', m'), (n, m))], \tag{7}$$

where $\hat{d}((n', m'), (n, m))$ is the weighted distance from the grid point (n', m') to the grid point (n, m). For example, for Type I local constraints and an asymmetric weighting function, n' and m' may take on any of the following values,

$$(n', m') \in \{(n-1, m-1), (n-1, m-2), (n-2, m-1)\} \tag{8}$$

and $\hat{d}((n', m'), (n, m))$ is given by

$$\hat{d}((n-1, m-1), (n, m)) = d(n, m), \tag{9a}$$

$$\hat{d}((n-1, m-2), (n, m)) = d(n, m), \tag{9b}$$

$$\hat{d}((n-2, m-1), (n, m)) = 2d(n, m). \tag{9c}$$

Thus the full DTW recursion for Type I local constraints and weighting function Type a is given by

$$D_A(n, m) = \min[D_A(n-1, m-1) + d(n, m), D_A(n-1, m-2) + d(n, m), D_A(n-2, m-1) + 2d(n, m)]. \tag{10}$$

Using the local optimality principle, a complete DTW algorithm is given by the algorithm

Step 1. Initialize $D_A(1, 1) = d(1, 1)\tilde{W}(1)$.
Step 2. Compute $D_A(n, m)$ recursively for $1 \le n \le N, \quad 1 \le m \le M$.
Step 3. $\hat{D} = D_A(N, M)/\mathbf{N}(\tilde{W})$.

This completes our review of the basic principles involved in applying dynamic programming to discrete word recognition. We will now describe the difficulties which arise when DTW algorithms are applied to connected word recognition problems and then we will show how the DTW principle can be modified for word spotting and connected word recognition applications.

2.2 Difficulties in connected word recognition

We shall assume that we are given a test pattern consisting of a sequence of connected words, spoken in a normal manner, for which the global beginning and ending points have been accurately located

and for which no further segmentation has been attempted. Given such a framework, the word spotting problem is to determine all subsections of the test pattern, if any, which match with a specified reference pattern, called the keyword. Thus, for word spotting a multiplicity of regions of the test pattern must be compared with the keyword pattern.

The connected word recognition problem, on the other hand, is to piece together reference patterns (obtained, in all our work, from isolated occurrences of words) to match the test pattern. The general approach to this problem will be the one proposed by Levinson and Rosenberg,[13] namely:

(*i*) Find the reference pattern that best fits a given section of the test pattern.

(*ii*) Use the position within the test pattern at which the best matching word ends to postulate the beginning of the following word.

(*iii*) Continue to concatenate reference patterns in this manner until the test pattern is exhausted.

Dynamic time-warping algorithms, as they have been applied to discrete word recognition applications, are not directly applicable to either the word spotting or the connected word recognition problem. There are two reasons why this is so. Figure 3 illustrates some of the problems which are encountered. In this figure we show the time

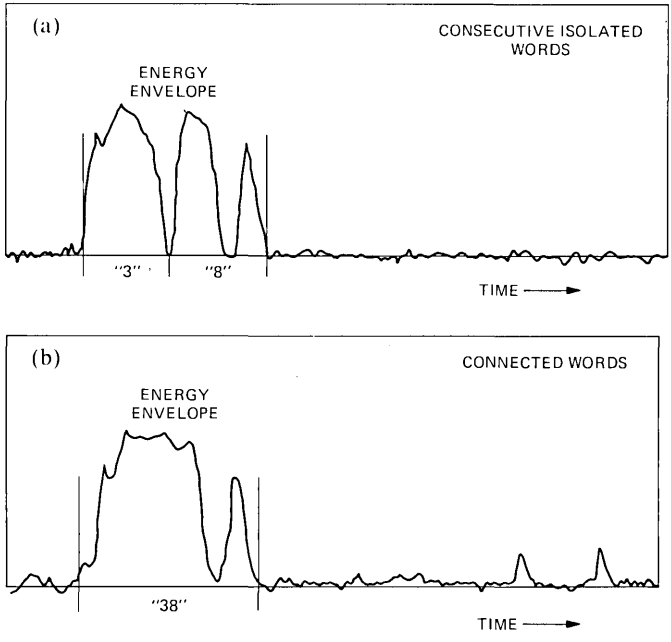


Fig. 3—Log energy for two speech utterances.

pattern for log intensity of two speech utterances, "3," "8" in part a, and "38" in part b. The utterance in part a was spoken with a discernible pause between the "3" and the "8," while the utterance in part b was spoken with no discernible pause between the "3" and the "8." Dynamic time-warping algorithms, as they have been applied to discrete word recognition, require a reliable set of word boundaries. However, as seen in Fig. 3b, a reliable segmentation for the utterance "38" is difficult, if not impossible, to obtain.

Another difficulty in using DTW algorithms, based on isolated word reference templates, for connected speech applications is the problem of coarticulation between words. For example, the final /*i*/ of the word "3" and the initial /e^i/ of the word "8" coarticulate strongly with each other. Thus, another fundamental assumption that has been relied on, namely that the characteristics of the isolated reference words which we are trying to match to our test utterance can be truly found in the test pattern, is not valid. In the next section we will describe the basic techniques that will be used to overcome these difficulties.

2.3 Basic approaches to connected speech recognition problems

In our approach to connected word recognition and word spotting we will make two changes from the structure of the isolated word DTW algorithm. One change is to no longer attempt to find the entire isolated reference pattern in the test pattern. We will still use isolated words as our reference patterns but will only expect a good match in the middle of the word, and not necessarily near the ends. Thus, we will not require that we be able to accurately match the beginning and ending points of the reference pattern to points within the test pattern. As a result, we would like to consider the possibility of overlapping reference patterns to recognize connected speech. In this manner we hope to account for both errors in the endpoint locations and for some of the gross features of coarticulation.

Another fundamental modification to the basic DTW algorithm is the use of beginning and ending *regions* rather than beginning and ending *frames*. In this manner we hope to avoid some of the problems inherent in requiring an accurate segmentation of the test utterance. Figure 4 defines, within a test pattern, a beginning region of size B (frames), with potential starting frames between b_1 and b_2 ($B = b_2 - b_1 + 1$), and an ending region of size E, with potential ending frames between e_1 and e_2 ($E = e_2 - e_1 + 1$). One possible DTW constraint would be that the best time-alignment contour may begin *anywhere* within the beginning region and end *anywhere* within the ending region. Three such potential paths are shown in Fig. 4. Such a framework would be used for word spotting, in which the beginning and ending regions correspond to the *entire* test pattern, or for connected word recogni-

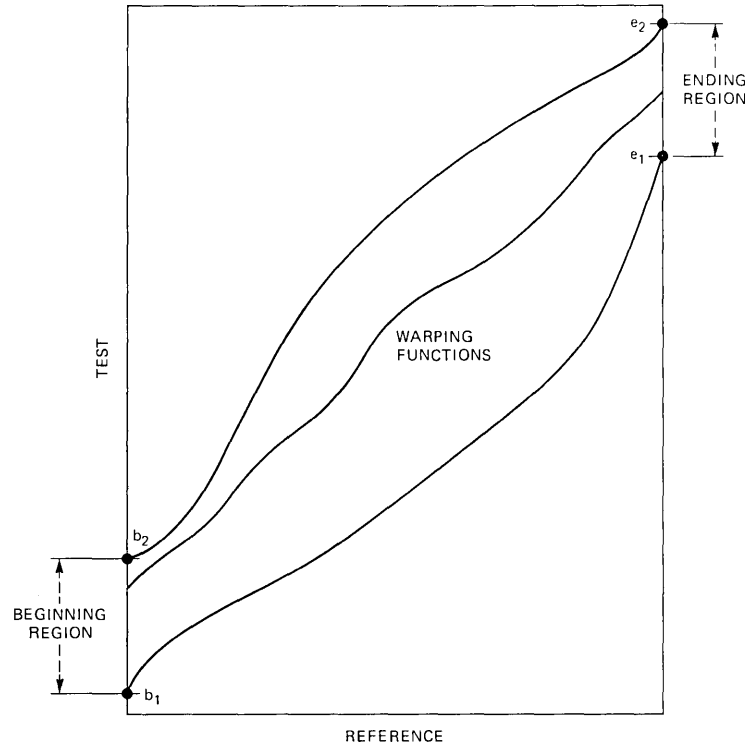


Fig. 4—Illustration of the use of beginning and ending regions.

tion, in which the ending region for one word is used to hypothesize the beginning region for the next word.

The use of beginning and ending regions modify the basic DTW algorithm by changing the constraints which are imposed on the ends of the time-alignment contour, i.e.,

$$i(1) = 1, \quad j(1) = b, \quad b_1 \leq b \leq b_2, \tag{11a}$$

$$i(K) = N, \quad j(K) = e, \quad e_1 \leq e \leq e_2. \tag{11b}$$

Thus, to find the optimal time-alignment contour, every possible beginning and ending point pair must be tried, that is,

$$\hat{D} = \min_{b_1 \leq b \leq b_2} \left[\min_{e_1 \leq e \leq e_2} \left[\min_{K, i(k), j(k)} [D(i(k), j(k)) \text{ s.t. } j(1) = b, j(K) = e] \right] \right]. \tag{12}$$

The amount of computation required to solve eq. (12) for the optimal path can be excessive, i.e., theoretically we require $B \cdot E$ separate time warps in the most general case. However, the amount of computation

required to solve eq. (12) may be reduced to a *single* time warp by judicious selection of the weighting function. If $\tilde{W}(k)$ is chosen to be the asymmetric weighting function, Type a ($\tilde{W}_a(k) = i(k) - i(k-1)$), and $\mathbf{N}(\tilde{W})$ is chosen appropriately ($\mathbf{N}(\tilde{W}_a) = N$), then $\hat{D}$ may be computed efficiently by a modified DTW algorithm as follows:

Step 1. Set $D_A(1, b) = d(1, b)$ for $b_1 \leq b \leq b_2$,

Step 2. Compute $D_A(n, m)$ recursively for $1 \leq n \leq N$, $b_1 \leq m \leq e_2$,

Step 3. $$\hat{D} = \frac{1}{N} \min_{e_1 \leq e \leq e_2} [D_A(N, e)].$$

This algorithm works because Step 1 initializes all possible beginning points, Step 2 computes the best path to a point (n, m) from any of the potential beginning points initialized in Step 1, and Step 3 finds the best possible ending point along any path from any possible beginning point. The particular choice of the asymmetric weighting function is important because its normalization factor is unaffected by the choice of the beginning or ending points, i.e., its normalization factor is always N. A dependence on the length of the test pattern, as in the symmetric weighting function, Type b, would require a separate time warp for each set of beginning and ending points because the effective length of the test $(e - b + 1)$ depends on the choice of the beginning and ending points.

An important factor, even with the savings of a single time warp, is the large amount of computation required for the DTW algorithm. Step 2 of the modified DTW algorithm is defined for $1 \leq n \leq N$, $b_1 \leq m \leq e_2$ and this region may be as large as $N \cdot M$. It is also not possible to significantly reduce this size by using restrictions on the slope of the warping contour when the ending region is left unspecified. This point is illustrated in Fig. 5, where the slope of the warping function is restricted to be between ½ and 2. We observe that, even with this restriction, when no ending region is specified, the area for which $D_A(n, m)$ must be computed is $\frac{3}{4}N^2 + B \cdot N$.

Two modifications to the DTW algorithm have been suggested to reduce this amount of computation. In particular, Sakoe and Chiba[3] have proposed that a time-warping path not be allowed to deviate significantly from a straight line, i.e., for any $i(k)$, the value of $j(k)$ is restricted such that

$$|j(k) - i(k) - \tilde{b} + 1| \leq R, \tag{13}$$

where $\tilde{b}$ is the center of the beginning region [$\tilde{b} = (b_1 + b_2)/2$] and R is the maximum deviation which is allowed. R must be chosen to at least cover the entire beginning region, i.e., $2R + 1 \geq B$. This algorithm

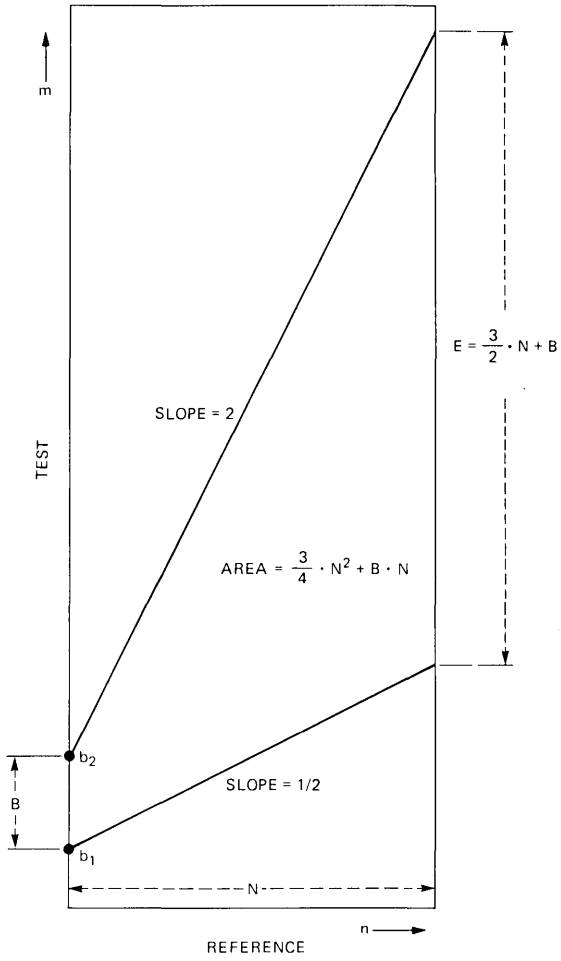


Fig. 5—Region of the (n, m) plane which is examined in a time warp for which no ending region is specified.

will be referred to as the *fixed range* DTW algorithm and is illustrated in Fig. 6a. Another range-reduction technique, proposed by Rabiner, Rosenberg, and Levinson[10] and described in detail by Rabiner and Schmidt[8] is shown in Fig. 6b. Here $j(k)$ is restricted to be within a fixed range about the best path so far, that is, the local minimum Formally, we have

$$|j(k) - c(k)| \leq \epsilon, \tag{14a}$$

$$c(k) = \underset{m}{\text{argmin}}[D_A(i(k) - 1, m)], \tag{14b}$$

$$c(1) = \tilde{b}, \tag{14c}$$

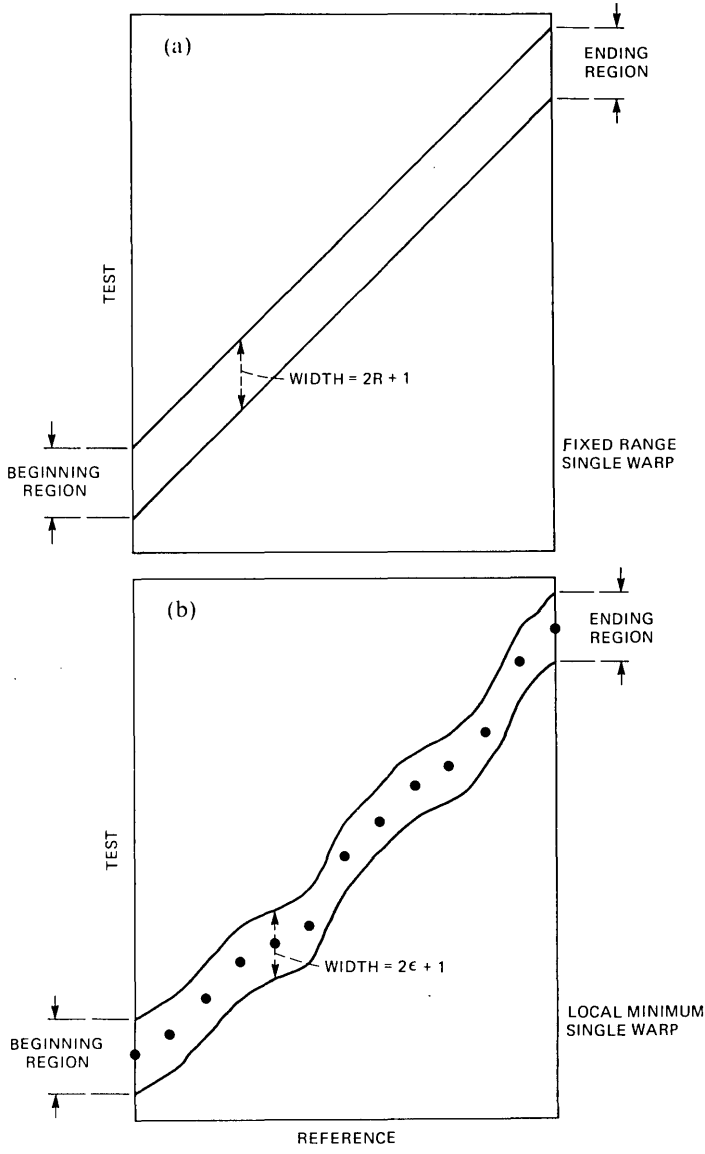


Fig. 6—Illustration of the fixed range and the local minimum DTW algorithms.

where $c(k)$ is the position, in the vertical direction, of the local minimum of $D_A(i(k) - 1, m)$, and ϵ is the allowable range about this local minimum. Thus, if $D_A(n, m)$ is computed in successive vertical strips, i.e., n is fixed and m is varied, then the range of one vertical strip is $\pm\epsilon$ about the local minimum of the previous vertical strip. This algorithm is referred to as the *local minimum* DTW algorithm.

Two fundamental differences exist between these two algorithms. The fixed range DTW algorithm, *a priori*, specifies the ending region from the specification of the beginning regions, i.e.,

$$E = 2R + 1, \tag{15a}$$

$$e_1 = \tilde{b} + N - R, \tag{15b}$$

$$e_2 = \tilde{b} + N + R, \tag{15c}$$

while the local minimum DTW algorithm defines the ending region implicitly from the local minimum of the last vertical strip, i.e.,

$$E = 2\epsilon + 1, \tag{16a}$$

$$e_1 = c(K) - \epsilon, \tag{16b}$$

$$e_2 = c(K) + \epsilon. \tag{16c}$$

The other fundamental difference between the two time-warping algorithms involves the number of time warps required to cover a beginning region. For the fixed range DTW algorithm the entire beginning region is most efficiently covered in a single time warp with $2R + 1 = B$, rather than several smaller time warps, because overlapping time warps may be merged together without loss of accuracy.

However, an analogous specification of the local minimum time-warping algorithm ($2\epsilon + 1 = B$) may not be truly optimal. Since one application of the local minimum DTW algorithm may follow only one local minimum path, erroneous decisions may be made because the true path may be "lost," i.e., the globally best path may not be within ϵ frames of the locally best path. As such, it may be better to try several smaller local-minimum time warps, thus allowing several different local-minimum paths to be tried, and to compare the results of these paths to determine the overall "best" path. Such a procedure is illustrated in Fig. 7. We assume that NTRY local minimum time warps are to be computed. Each time warp has (about its respective local minimum) a local range of $\pm\epsilon$ and the centers of two adjacent time warps are initially separated by δ. The entire region covered by the NTRY time warps is given by

$$\Delta = 2\epsilon + 1 + (\text{NTRY} - 1) \cdot \delta. \tag{17}$$

To cover the entire beginning region, NTRY, ϵ and δ are chosen so that $\Delta = B$.

In the next section of this paper we describe experiments designed to measure the relative strengths and weaknesses of the fixed range and the local minimum DTW algorithms and also to determine reasonable choices for the parameters δ, ϵ, and NTRY for both word spotting and connected word recognition applications.

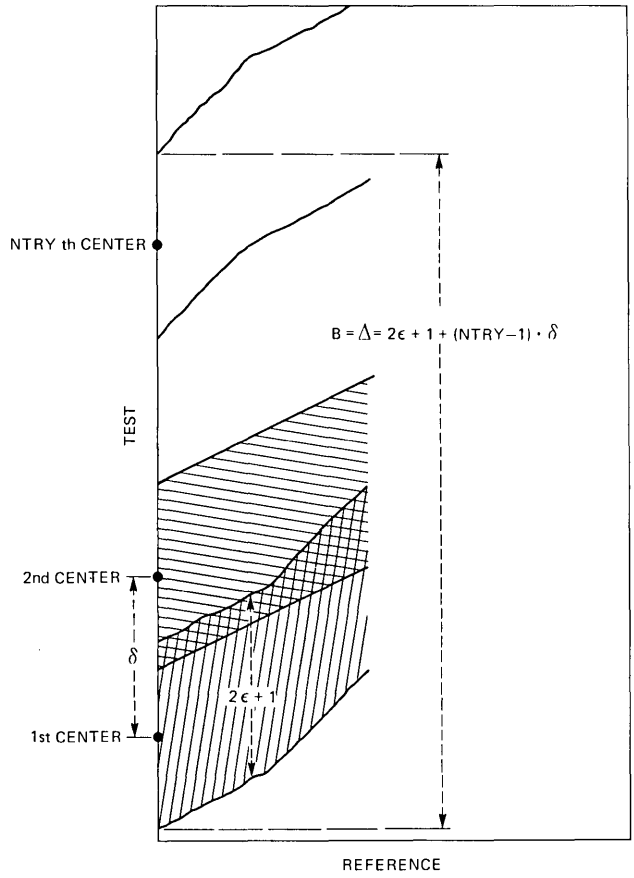


Fig. 7—Illustration of the parameters of the local minimum DTW algorithm.

III. EXPERIMENTS IN DYNAMIC TIME WARPING FOR CONNECTED SPEECH RECOGNITION

This section presents the results of experiments designed to compare the fixed range and the local minimum DTW algorithms. We also describe the results of several experiments designed to study the parameters of the local minimum algorithm. Finally, we show how these results may be applied to the problems of word spotting and connected word recognition.

3.1 Comparison of the time warping algorithms

In our initial experiment the recognition accuracies achieved by both the fixed range and the local minimum DTW algorithms for a modified *isolated* word recognition problem are compared. The test utterances consisted of 54 words from a vocabulary of computer terms,

spoken by each of 4 talkers, for a total of 216 utterances. The test utterances were recorded over a dialed-up telephone line, band-limited to 3.2 kHz, digitized at 6.67 kHz, and analyzed every 15 ms with an eighth-order LPC analysis using a 45-ms window (i.e., successive frames overlapped by 30 ms). Local distance scores, $d(i(k), j(k))$,were calculated using Itakura's log likelihood ratio.[4] The reference patterns consisted of two templates per word of the vocabulary formed by a speaker-independent clustering technique.[14]*

To evaluate the relative performance of the two DTW algorithms the test utterances were modified so that a beginning region could be specified as some range about the true beginning point. No ending region was specified. For the sake of comparison, R and ϵ were both set equal to eight frames† and NTRY was set to one. Figure 8 shows the recognition results for both algorithms as a function of the four different local constraints (used in the DTW algorithms) defined in Section 2.1. We observe that the local minimum DTW algorithm performed better than the fixed range DTW algorithm for *all* local constraints.

In another comparison we generated ten pseudo-connected test sequences by artificially embedding (at an arbitrary frame) an isolated digit into a connected digit sequence, both uttered by the same talker. We then used both DTW algorithms to "spot" the embedded digit using two speaker-dependent templates per digit. The parameters of the two DTW algorithms that were used were the same ones as in our initial experiment ($\epsilon = 8$, $R = 8$). To spot the embedded digit, every possible beginning region of size $2\epsilon + 1$ ($= 2R + 1$) was tried. The number of times that the DTW algorithm found the (correct) best path (as determined by the lowest overall distance achieved by any beginning region) was recorded. We also recorded the ending point of the embedded word, as estimated by the word spotting procedure. Results showed that both the local minimum and the fixed range DTW algorithms were able to locate the endpoint of the embedded word with a high degree of accuracy. (The average error between the true ending frame and the estimated ending frame was 1.2 frames for both DTW algorithms.)

Figure 9 shows the relative performance of the two DTW algorithms for this simple word spotting experiment. These figures plot the number of times that the particular DTW algorithm found the proper path (as determined by the lowest-distance score achieved) for each of

---

* The speaker-independent reference template set was a subset of the 12 template per word set used in Ref. 14. This modification was used to reduce computation (and hence reduce accuracy somewhat). For the purpose of our experiments (i.e., the relative comparison of the fixed range and the local minimum DTW algorithms) this modification was of little consequence.

† Setting R and ϵ equal is a fair comparison of the two methods since the computation is the same for both methods.

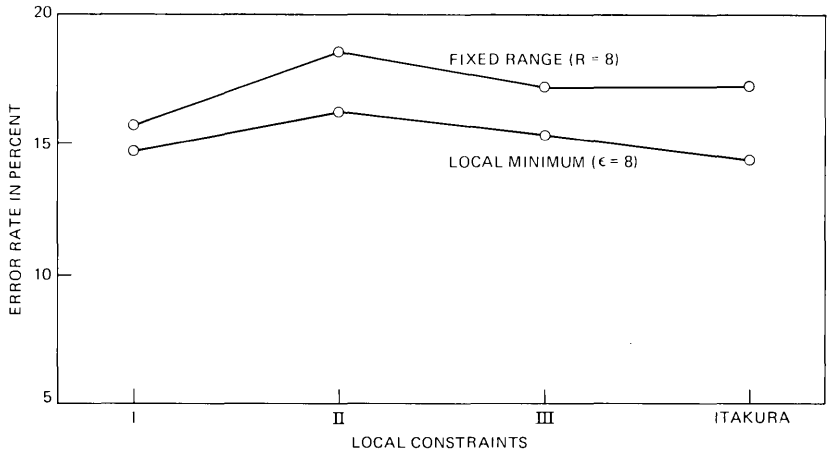


Fig. 8—Results for word recognition using both the fixed range and the local minimum DTW algorithms.

the ten embedded digits. We observe from Fig. 9 that the local minimum DTW algorithm found the best path more often than the fixed range DTW algorithm for almost all digits.

We also observe that the local minimum algorithm was able to find the best path 17 times (the maximum number possible, $2\epsilon + 1$) for 8 of the 10 digits, while the fixed range algorithm never achieved this accuracy.

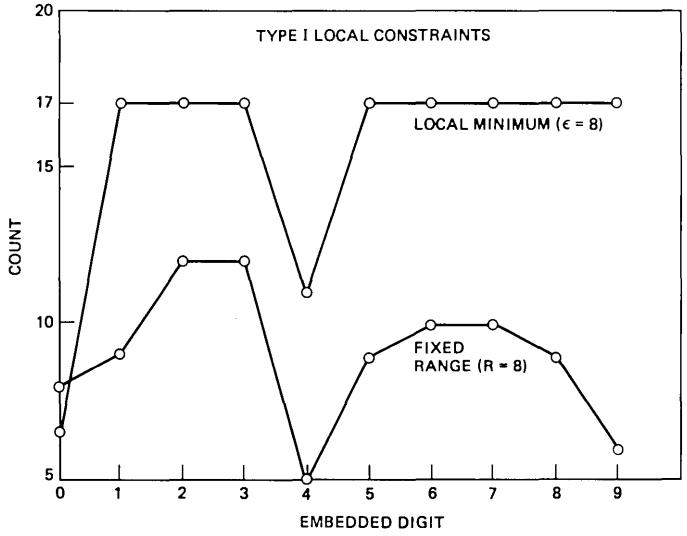


Fig. 9—Results for word spotting using both the fixed range and the local minimum DTW algorithms.

The results of these two sample experiments showed that the local minimum DTW algorithm performed consistently better than the fixed-range DTW algorithm. In the next section we describe experiments designed to more fully study some of the parameters of the local-minimum time-warping algorithm.

3.2 *Examination of the parameters of the local-minimum dynamic time-warping algorithm*

To understand the effects of the various combinations of the parameters Δ, δ, NTRY, and ϵ on the performance of the local minimum DTW algorithm, a series of connected digit-recognition experiments was performed. A total of 80 strings of from 2 to 5 connected digits each (20 strings of each length) were recorded by each of the two talkers. These strings were the same as those used by Rabiner and Schmidt.[8] In the recognition task we used two speaker-dependent templates per digit. The first step in the experiment was to "spot" the ending point of the first digit in each string via a local-minimum algorithm ($\epsilon = 11$, NTRY = 1) using the known beginning point of the first digit. Then an attempt was made to recognize the second digit in the string. Because of inaccuracies in "spotting" the ending point of the first digit, and because of coarticulation effects, it was not possible to precisely determine the beginning point of the second digit, and, as such, a beginning region for the second digit was centered around the ending frame of the first digit, as determined by the "spotting" procedure. The best candidate for the second digit was chosen as that template which achieved the lowest overall average distance, regardless of where it ended. Several values of ϵ, δ, Δ, and NTRY were used and the accuracies and distance scores for the recognition of the second digit were recorded.

Figure 10 shows, for a large value of Δ (27 in this case), the average best distance score for all NTRY time warps as a function of δ, for several values of ϵ. Two curves are shown in each part of the figure. The solid curve is the case when the reference word is the same as the second word in the test strings. The dashed curve represents the case in which the reference is different from the second word in the test string. Examination of Fig. 10 shows that the average best distance for both "same words" and "different words" increases as δ increases. However, we observe that when the reference is different from the second digit in the test utterance (i.e., the dashed curves), the average distance generally increases as δ increases, but, when the reference and the test words are the same (i.e., the solid curves), the average best distance is constant for small values of δ and increases only beyond the critical value $\delta = 2\epsilon + 1$. This critical value, $\delta = 2\epsilon + 1$ (shown by a caret in the scales of Fig. 10), is a particularly important value of δ

because for $\delta < 2\epsilon + 1$, consecutive time warps overlap in their beginning regions, and for $\delta > 2\epsilon + 1$ there are frames between two consecutive time warps which are not covered by either beginning region. When $\delta = 2\epsilon + 1$, we have the case where there is no overlap in adjacent beginning regions and no skipped frames between these regions. From the results shown in Fig. 10 we conclude that, on average, there is no loss in performance in the local-minimum DTW algorithm as long as no potential beginning frames are skipped, i.e., as long as $\delta \leq 2\epsilon + 1$.

One explanation of why δ may be taken as large as $2\epsilon + 1$, i.e., no overlapping of beginning regions, without an appreciable loss of accuracy, is shown in Fig. 11. Here we show the progress of a set of typical paths in which the starting regions overlap. By the nature of the local-minimum DTW algorithm, best paths from overlapping time warps tend

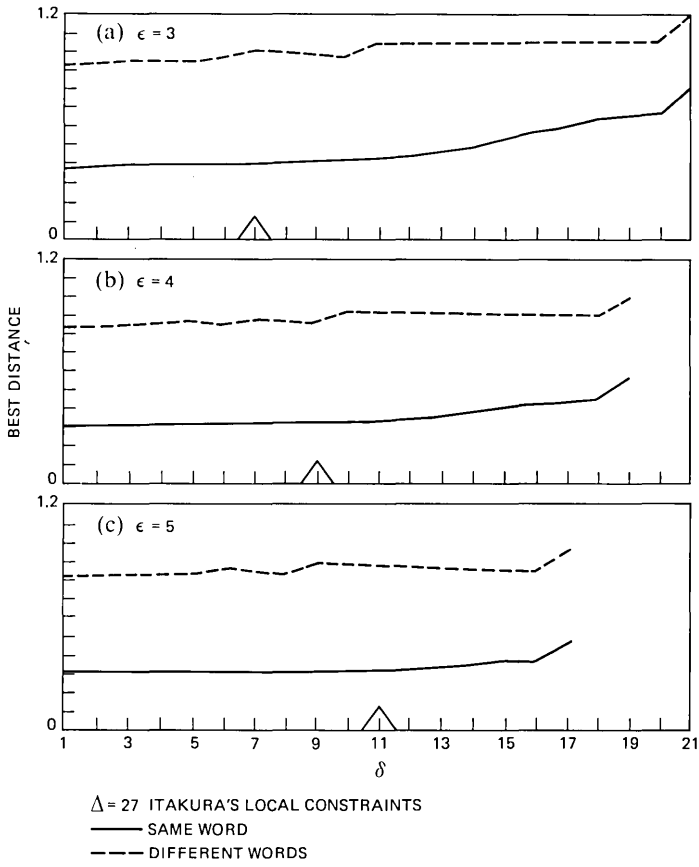


Fig. 10—Distance scores for the local minimum DTW algorithm as applied to connected digit recognition.

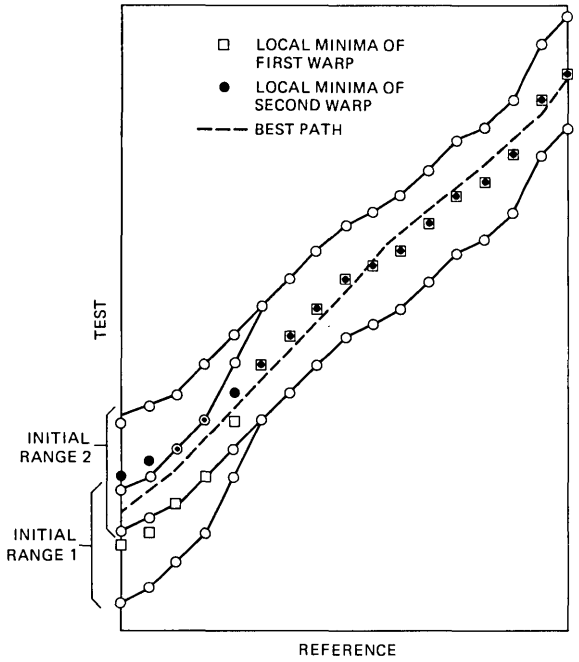


Fig. 11—Illustration of path merging for two adjacent local-minimum time warps.

to merge if there is a good path common to both of their beginning regions. Figure 12 shows the effects of path merging (of the local minimum DTW algorithm) on the digit recognition accuracies. Here we plot the recognition error rate for the second digit in the test sequences as a function of δ for various values of ϵ. We see that, for a fixed ϵ, it is possible to increase δ with essentially no loss in accuracy as long as $\delta \leq 2\epsilon + 1$.*

Figure 12 also shows that $\epsilon = 6$ provides the minimum error rate. It is reasonable to expect that as ϵ is made too small, good paths may easily become lost; but as ϵ is made too large, incorrect paths may start to generate low scores and thus cause errors. Thus, a finite value of ϵ is probably optimum. Unfortunately, such a value will have to be determined for each application.

Another interesting effect on recognition accuracy for various combinations of ϵ, δ, Δ, and NTRY is shown in Fig. 13. Here we plot recognition error rates for the second digit of our test utterances for two cases, namely $\epsilon = (\Delta - 1)/2$ (NTRY = 1), and for the best combination of ϵ, δ, and NTRY (as determined by the lowest-recog-

* Note that for Δ fixed, the largest possible δ is $\delta = \Delta - 2\epsilon - 1$ (NTRY = 2) so that the curves for the various values of ϵ in Figure 12 are defined only for those values of δ such that $\delta \leq \Delta - 2\epsilon - 1$.

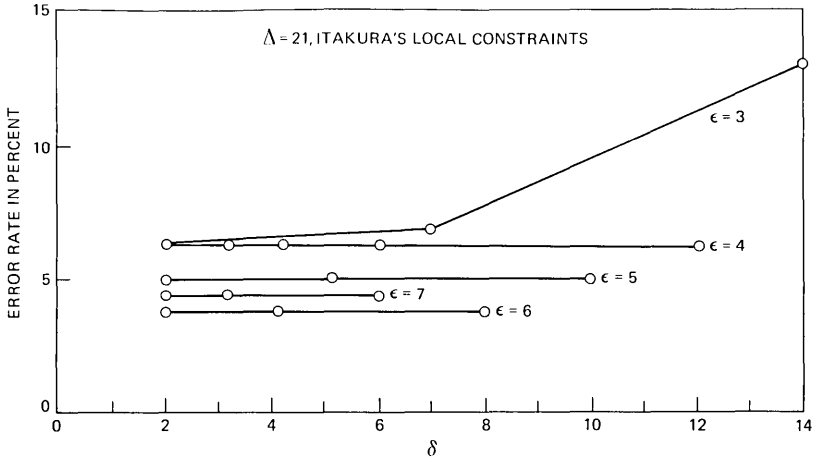


Fig. 12—Digit error rate for connected digit recognition using the local minimum DTW algorithm and several values of ϵ and δ.

nition error rate). We see that, for smaller values of Δ, a single warp performs as well as any combination of ϵ, δ, and NTRY, and as Δ increases, the differences in error rates between the best possible ϵ, δ, and NTRY combination and a single warp remains less than 2.5%. Thus, it might be possible to perform some type of connected word recognition using only a single local-minimum time warp per word. In the next section we describe how the results of our experiments have actually been applied to both word spotting and connected word recognition applications.

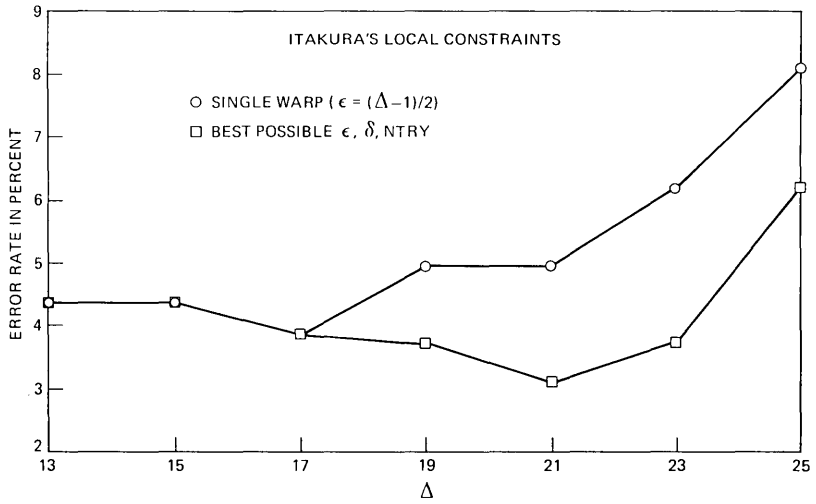


Fig. 13—Digit error rates for connected digit recognition using the local minimum DTW algorithm.

3.3 Application of DTW algorithms to word spotting and connected word recognition

We have shown that, both for connected word recognition and word spotting applications, the local minimum DTW algorithm performs consistently better than the fixed range DTW algorithm. We have also shown that, given a value of ϵ, δ may be chosen as large as $\delta = 2\epsilon + 1$ without significant degradation in the performance of the local minimum DTW algorithm. Since, for a fixed beginning region (i.e., a fixed Δ), the number of time warps is given by NTRY $= 1 + (\Delta - 2\epsilon - 1)/\delta$, the best choice for δ is $\delta = 2\epsilon + 1$. This minimizes the number of time warps which need to be performed. For the problem of word spotting the obvious choice for Δ is $\Delta = M$, i.e., the entire length of the test pattern. For this case optimal values of ϵ and NTRY must still be determined. In general, the selection of ϵ and NTRY depends on several factors. As ϵ is increased, the chance of a missed keyword decreases because more paths are examined, but the chance of a false alarm increases. Also, as ϵ increases, the value of NTRY decreases [NTRY $= \Delta/(2\epsilon + 1)$ for $\delta = 2\epsilon + 1$], thereby reducing the amount of computation required. Thus, misses, false alarms, and the amount of computation must be traded-off in the selection of ϵ and NTRY for a word spotting application.

In a connected word recognition application, however, we not only must choose ϵ and NTRY but must also choose Δ. We have shown that for $\Delta \leq 17$ frames, it is possible to do connected digit recognition using only a single local-minimum time warp per word. However, we also found that the best recognition accuracy was achieved with $\Delta =$ 21 but not with a single local-minimum time warp. Thus, there is an apparent trade-off between recognition accuracy and speed of computation. However, work by Rabiner and Schmidt[8] has shown that it is better not to center the beginning region of one word around the end of the previous word, as we did, but, rather, to center the beginning regions of one word several frames earlier than the ending region of the previous word. The reason for this is that the isolated reference patterns tend to be longer than the spoken connected words, and thus, the time warps tend to overestimate the ending frame of each word. We tried a simple experiment in which the beginning region of one word was centered eight frames earlier in the test pattern than the end of the previous word. The values of ϵ, NTRY, and Δ were $\epsilon = 8$, NTRY $= 1$, and $\Delta = 17$. Using these values and the same test utterances used by Rabiner and Schmidt,[8] i.e., 80 sequences of from 2 to 5 digits each spoken once by each of six talkers, we achieved a string recognition rate of 429 correct strings out of 480 possible. This may be compared with a total of 442 correct strings using $\epsilon = 8$, $\delta = 3$, and NTRY $= 4$, as reported by Rabiner and Schmidt. It should be noted,

however, that the system of Rabiner and Schmidt used multiple candidate strings while our simple experiment did not. When we reran the system of Rabiner and Schmidt using only a single candidate string ($\epsilon = 8$, $\delta = 3$, NTRY = 4) we found only 430 correct strings out of the 480 possible. Thus, with a single local-minimum time warp per word we achieved results comparable to those achieved by the use of four local-minimum time warps per word.

IV. CONCLUSIONS

We have shown that dynamic time warping algorithms can be efficiently applied to both word spotting and connected word recognition. We have demonstrated the relative performance superiority of the local minimum DTW algorithm over the fixed-range DTW algorithm. It was also shown that the beginning regions of successive applications of the local minimum DTW algorithm need not overlap to achieve accuracy comparable to overlapping beginning regions. We have found that, for small beginning regions (small Δ), a single local-minimum time warp [with $\epsilon = (\Delta - 1)/2$, NTRY = 1] was as accurate as (and more computationally efficient than) any combination of the parameters ϵ, δ, and NTRY. Finally, we found that an extremely simple connected digit recognition system, i.e., a single local-minimum time warp per word using only one candidate string, achieved a string recognition rate of nearly 90 percent.

REFERENCES

1. T. B. Martin, "Practical Applications of Voice Input to Machines," Proc. IEEE, *64* (April 1976), pp. 487–501.
2. G. M. White and R. B. Neely, "Speech Recognition Experiments with Linear Prediction, Bandpass Filtering and Dynamic Programming," IEEE Trans. Acoust. Speech, Signal Proc., *ASSP-24* (April 1976), pp. 183–8.
3. H. Sakoe and S. Chiba, "A Dynamic Programming Approach to Continuous Speech Recognition," Proc. Int. Congress Acoustics, Budapest, Hungary, 1971, Paper 20C-13.
4. F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," IEEE Trans. Acoust. Speech, Signal Proc., *ASSP-23* (February 1975), pp. 67–72.
5. J. S. Bridle, "An Efficient Elastic Template Method for Detecting Given Words in Running Speech," Proc. British Acoust. Soc. Meetings, London, England, April 1973, Paper 73SHC3.
6. R. W. Christiansen and C. K. Rushforth, "Detecting and Locating Keywords in Continuous Speech Using Linear Preductive Coding," IEEE Trans. Acoust. Speech, Signal Proc., *ASSP-25* (October 1977), pp. 361–7.
7. H. Sakoe, "Two-Level DP Matching—A Dynamic Programming Based Pattern Matching Algorithm for Connected Word Recognition," IEEE Trans. Acoust. Speech, Signal Proc., *ASSP-27* (December 1979), pp. 588–95.
8. L. R. Rabiner and C. E. Schmidt, "Application of Dynamic Time Warping to Connected Digit Recognition," IEEE Trans. Acoust. Speech, Signal Proc., *ASSP-28* (August 1980).
9. C. S. Myers and L. R. Rabiner, "A Level Building Dynamic Time Warping Algorithm for Connected Word Recognition," IEEE Trans. Acoust. Speech, Signal Proc., to appear.
10. H. Sakoe and S. Chiba, "Dynamic Programming Optimization for Spoken Word

Recognition," IEEE Trans. Acoust. Speech, Signal Proc., *ASSP-26* (February 1978), pp. 43–9.
11. L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson, "Considerations in Dynamic Time Warping for Discrete Word Recognition," IEEE Trans. Acoust. Speech, Signal Proc., *ASSP-26* (December 1978), pp. 575–82.
12. C. S. Myers, L. R. Rabiner, and A. E. Rosenberg, "Performance Tradeoffs in Dynamic Time Warping Algorithms for Isolated Word Recognition," IEEE Trans. Acoust. Speech, Signal Proc., to appear.
13. S. E. Levinson and A. E. Rosenberg, "A New System for Continuous Speech Recognition—Preliminary Results," Proc. Int. Conf. Acoust. Speech, Signal Proc. (April 1979), pp. 239–44.
14. L. R. Rabiner and J. G. Wilpon, "Speaker Independent, Isolated Word Recognition for a Moderate Size (54 Word) Vocabulary," IEEE Trans. Acoust. Speech, Signal Proc., *ASSP-27* (December 1979), Part I, pp. 583–7.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 3, March 1981
Printed in U.S.A.

The Material Dispersion Zero in Infrared Optical Waveguide Materials

By K. NASSAU

(Manuscript received October 3, 1980)

The wavelength where the material dispersion is zero, i.e., that wavelength for which a multimode lightguide would function at highest bandwidth, can be estimated using fundamental materials properties. Operation of lightguides at or near this wavelength is essential for the use of very long fiber lengths with high bandwidth. Calculations have been performed for large groups of metal oxides, fluorides, chlorides, and bromides. These data can be used to locate materials with potential for ultralow-loss fibers at longer wavelengths than are currently in use.

I. INTRODUCTION

Operation of optical waveguides at wavelengths up to 1.3 μm and 1.6 μm is being actively investigated and losses of a few tenths dB/km appear to be feasible. Operation at a longer wavelength has the attraction that much lower losses should be possible since the λ^{-4} relationship controls the scattering loss, subject of course to the achievement of adequately low intrinsic and impurity absorptions. This should permit large distances between repeaters.

The utilization of very long lengths of low-loss fibers is critically dependent on the material dispersion parameter $(\lambda/c)(d^2n/d\lambda^2)$, where n is the refractive index at wavelength λ and c is the velocity of light. At the wavelength λ_0 where the material dispersion is zero, the delay distortion in multimode fibers is minimized and maximum bandwidth can be achieved. In single-mode fibers, zero total dispersion requires balancing the waveguide dispersion with the material dispersion; the maximum bandwidth then occurs at a wavelength longer than λ_0.

Several reports have discussed possible longer wavelength fiber materials, including the extended discussion of Goodman[1] for the 4-μm band, the polycrystalline materials Tl(Br,I) of Pinnow et al.[2] and Ag(Cl,Br) of Garfunkel et al.,[3] and $ZnCl_2$ glass of Van Uitert and

Wemple.[4] The use of fluoride glasses based on ZrF_4, HfF_4, and AlF_3,[5,6] mainly derived from earlier work by Poulain and co-workers (e.g., Refs. 7 and 8) has also been studied. A limited consideration of a few[1] or only a single[4] value of the material dispersion was given in some of these reports, while others[2,3,5,6] ignored this parameter.

Wemple[9] has given a general technique for calculating an approximate value of the wavelength of the material dispersion zero, based on materials parameters which were discussed in detail in his previous report.[10] Using this approach it is possible to calculate λ_0 values for large groups of potential optical waveguide materials and thus locate the most promising candidates for operation at desired wavelengths. In this study we focus our attention on metal oxides, fluorides, chlorides, and bromides which may have such a potential and from which glasses might be made. The reasons for concentrating on these four groupings and on glasses specifically, as well as further limiting factors, are discussed in the appendix.

These data can, of course, be used to direct the investigation of specific materials with potential for ultralow loss at longer wavelengths than are currently in use. They can also eliminate from consideration some already studied materials which may be transparent in this region, but which have too low a λ_0 value.

II. THE CALCULATION OF λ_0

Based on considerations of optical oscillator strengths and excitation energies, Wemple[9,10] has developed a formalism from which λ_0, the wavelength for which $d^2n/d\lambda^2 = 0$ is given as

$$\lambda_0 = hc\left(\frac{10^{-10}cnf\mu d^3}{4\pi e^2 ZE^3}\right)^{1/4}, \tag{1}$$

where E is the average electronic (Sellmeier) excitation gap (usually a few eV higher than the band gap), f is the normalized oscillator strength, Z is the formal valence of the anion A, n is the number of valence electrons on the anion in the compound, d is the anion–cation distance, μ is the reduced mass of the anion–cation pair, h is Planck's constant, c is the velocity of light, and e is the charge on the electron.

Most often $n = 8$ for closed-shell anions, and eq. (1), rewritten in practical units, then reduces to

$$\lambda_0 = 2.96\left(\frac{d^2f\mu}{E^3Z}\right)^{1/4}, \tag{2}$$

where λ_0 is in μm, d is in Å, and E and f are in eV. Here $Z = 1$ for halides, 2 for oxides and chalcogenides, and 3 for pnictides; the reduced mass for a compound written as AB_b (i.e., with one cation A and with

b not necessarily an integer) of atomic weights M_A and M_B is given by

$$\mu = \frac{M_A M_B}{M_A + bM_B}. \tag{3}$$

For compound glasses, eq. (4) can be used to determine values for use in eqs. (2) and (3) for a mixture of i components, containing x mole fraction each of PR_r where P is a cation, R is an anion, and r is the number of anions per cation (not necessarily integral) with $\sum_i x_i = 1$:

$$\begin{aligned} f &= \sum x f_{PR_r}, \\ E &= \sum x E_{PR_r}, \\ Z_A &= \sum x Z_{PR_r}, \\ M_A &= \sum x M_P, \\ b &= \sum x r, \\ M_B &= \frac{1}{b} \sum x r M_R, \\ d &= \frac{1}{b} \sum x r d_{PR}. \end{aligned} \tag{4}$$

In these equations intensive properties, such as energies, are averaged according to the composition, but extensive properties, such as the bond lengths and the effective cation masses, are additionally weighted by the number of anion bonds per cation; b is the composition-weighted average of this last parameter.

Values of the required parameter were taken from Wemple's collection of data[10] when available or from the literature. Otherwise estimates for E and f (to the nearest ½ eV and so listed in Tables I and II) were made following Wemple's principles, allowing for coordination number, bond length anomalies, ionicity, anion contact, and shallow electron cores.[10] This last factor is significant for the d^{10} core of the Ag halides where N is effectively 14 rather than 8, introducing a correction factor for λ_0 in eq. (2) of $(14/8)^{1/4} = 1.15$; smaller corrections apply elsewhere, for example to the s^2 cores of the Tl and Pb halides.

Wemple has also pointed out that although amorphous nonvitreous solids differ significantly from crystals in the applicability of the approach used, glasses do not; they can be viewed as loosely-packed versions of the crystals to a good approximation.[10]

III. VALUES OF λ_0 FOR OXIDES, FLUORIDES, CHLORIDES, AND BROMIDES

The E and f values used and the λ_0 wavelengths obtained are listed for oxides in Table I and for fluorides, chlorides, and bromides in Table

Table I—Parameters for the calculation of λ_0 for oxides

Oxide	E	f	λ_0	Oxide	E	f	λ_0
MoO_3	6	4	2.8	Sc_2O_3	11	3	1.8
WO_3	6	4	3.0	Y_2O_3	9	4	2.4
P_2P_5	13.5	5	1.3	La_2O_3	8	4	2.9
As_2O_5	11	5	1.8	ZnO	6.1	3.1	2.7
Nb_2O_5	7	4	2.6	PbO	4.7	3.8	3.9
Ta_2O_5	7	4	2.7	BeO	13.7	5.1	1.2
SiO_2	13.3	5	1.3	MgO	11.4	2.8	1.6
GeO_2	11.0	4.0	1.7	CaO	9.9	2.8	2.0
SnO_2	8.1	3.0	2.2	SrO	8.3	2.5	2.5
TeO_2	6.3	3.9	2.8	BaO	7.1	2.4	3.0
TiO_2	5.5	3.8	2.8	Tl_2O	4	4	5.3
ZrO_2	11	4	2.0	Li_2O	12	5	1.6
HfO_2	10	4	2.2	Na_2O	11	4	2.1
ThO_2	10	4	2.3	K_2O	11	4	2.3
B_2O_3	12.4	6.8	1.3	Rb_2O	10	3	2.5
Al_2O_3	13.4	3.8	1.4	Cs_2O	10	3	2.7
Ga_2O_3	9½	4	2.0				
In_2O_3	7	4	2.8	GaAs	3.7	4.5	6.3
As_2O_3	11	5	1.9	ZnTe	4.4	5.2	6.6
Sb_2O_3	7	5	2.9	BN	10.6	4.5	1.2
Bi_2O_3	5	5	3.9				

Table II—Parameters for the calculation of λ_0 for halides

	X = F			X = Cl			X = Br		
	E	f	λ_0	E	f	λ_0	E	f	λ_0
LiX	16.5	3.8	1.3	11.0	4.8	2.3	9.5	4.8	2.7
NaX	15.1	2.8	1.7	10.5	3.5	2.9	9.1	3.7	3.7
KX	14.7	3.1	2.1	10.5	3.1	3.3	9.2	3.2	4.2
RbX	14	3½	2.4	10.5	3.3	3.7	9.3	3.3	4.8
CsX	13.5	4.9	3.3	10.6	3.5	4.1	9.4	3.4	5.3
AgX*	10½	3	3.0	7	3	5.1	5½	3.1	6.7
TlX*	9	3	3.5	5.5	2.9	6.6	5.2	3.3	8.5
BeX_2	16	5	1.2	11	5	1.8	9½	5	2.1
MgX_2	16.8	3.9	1.4	11	4	2.4	9½	4	2.9
CaX_2	15.7	3.1	1.7	10½	3	2.7	9	3	3.4
SrX_2	15	3	1.9	10½	3	3.1	9	3	4.0
BaX_2	13.8	3.1	2.3	10½	3	3.5	9	3	4.5
ZnX_2*	13	3	2.1	9	3	3.5	7½	3	4.6
CdX_2	11	3	2.3	7½	3	4.0	6	3	5.4
HgX_2	9	3	2.8	—	—	—	—	—	—
SnX_2*	10	3	2.6	7	3	4.4	5½	3	6.1
PbX_2*	8.4	2.8	3.3	6½	3	5.1	5	3	7.4
ScX_3	16	3	1.4	10	3	2.6	8½	3	3.2
YX_3	16	3	1.6	10	3	2.9	8½	3	3.7
LaX_3	16	3	1.8	10	3	3.3	8½	3	4.3
LuX_3	16	3	1.7	—	—	—	—	—	—
AlX_3	16	3	1.2	—	—	—	—	—	—
GaX_3	15	3	1.5	—	—	—	—	—	—
InX_3	14	3	1.8	7	3	4.4	5½	3	5.3
TlX_3	13	3	2.0	—	—	—	—	—	—
BiX_3*	8	3	3.2	6½	3	4.8	5	3	6.9
TiX_4	14	3	1.5	—	—	—	—	—	—
ZrX_4	14	3	1.7	10	3	2.7	9	3	3.3
HfX_4	14	3	1.8	10	3	3.0	9	3	3.8
ThX_4	14	3	1.9	10	3	3.2	9	3	4.0
SnX_4	13	3	1.8	—	—	—	—	—	—
PbX_4	13	3	1.9	—	—	—	—	—	—

* Corrections applied for shallow electron cores.

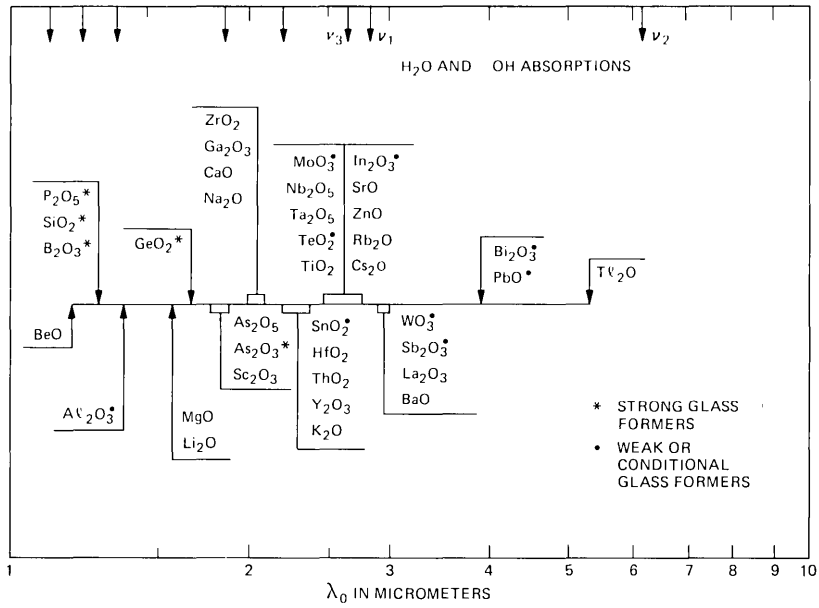


Fig. 1—Material dispersion crossover wavelengths for oxides. Also shown are the ν_1, ν_2, and ν_3 fundamental water absorptions; "water" in fused silica shows ν_3 as well as the indicated absorptions at lower wavelengths.

II; the more readily accessible values of d and μ are omitted. The results are graphically presented in Figs. 1 and 2. A definite precision cannot be given for λ_0; we feel that most values are accurate to ±10 percent, although in a few instances the uncertainty may well be a little bigger. Note that since the fourth root is involved in eq. (2), the results are only moderately sensitive to d and E and even less so to f and μ.

Most oxides with a value of E less than $4\frac{1}{2}$ eV were excluded, since such substances are deeply colored and would also be expected to show significant intrinsic absorption in the wavelength region of interest. An exception was made with Tl_2O (black, $E \approx 4$ eV), which is often an ingredient in glasses, in small amounts, since it reduces the melt viscosity and assists in the removal of bubbles. Oxides were also generally excluded, which would be expected to decompose under conditions required for glass melting. The oxides which are strong glass formers are marked, as are those which are weak or conditional glass formers, i.e., those which do not form glass by themselves but will do so in combination with other similar compounds.[11]

The halides of Table II and Fig. 2 were selected in an analogous way. Here BeF_2 and $ZnCl_2$ are the only strong glass formers; the

occurrence of glasses based on BeF_2 and other fluorides has been summarized by Sun.[12] Glasses based on the weakly glass forming ZrF_4, HfF_4, and AlF_3 have been studied by Poulain and others.[5-8]

Figure 1 also includes the fundamental vibrations of the free H_2O molecule,[13] marked ν_1, ν_2, and ν_3, as well as the positions of the dominant "water," i.e., OH, absorptions in fused silica, consisting of ν_3 and its overtones, and combinations with SiO_4 vibrations.[14] These absorptions tend to become more intense and broader with increasing wavelength. In view of the difficulty of eliminating all traces of water, operation in windows between these absorptions may be desirable. For fused silica these windows occur at about 1.3, 1.6, and 2.5 μm; since SiO_4 vibrations are involved, shifts in both position and intensity must be expected in moving to different systems, although differences may not be large with related oxides, such as GeO_2. With halides, however, significant difference may exist.

Another factor sometimes considered is the location of the windows in the atmosphere occurring at about 1.3, 1.6, 2.3, 3.5, to 4, and 8 to 13 μm.[15] It was primarily this consideration that caused Goodman[1] to limit his discussion to materials for the 4-μm band. This does not seem to be a meaningful limitation, however, since the path of optical waveguide communication systems does not normally include any atmospheric links.

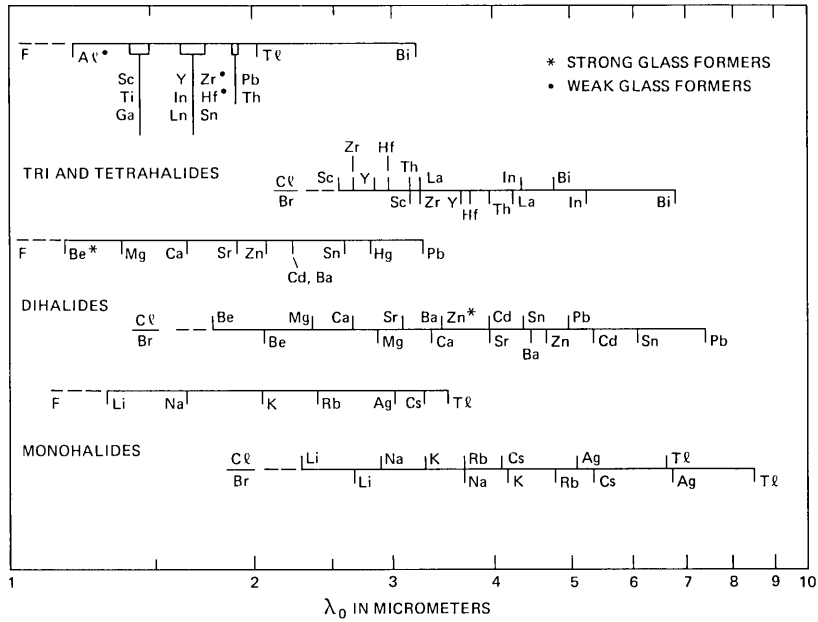


Fig. 2—Material dispersion crossover wavelengths for fluorides, chlorides, and bromides.

IV. DISCUSSION

In the case of the oxides of Table I and Fig. 1, λ_0 extends only to about 5 μm, with very limited choice above 3 μm. The use of chalcogenides, such as ZnTe, and pnictides, such as BN and GaAs, also included for reference in Table I, would extend λ_0 to much longer wavelengths for heavier anions, but the reduced bandgap of such compounds would also raise the intrinsic absorption (also see the appendix).

The only strong glass formers among halides are BeF_2 and $ZnCl_2$,[11] the former being highly toxic and the latter hygroscopic. Figure 2 shows a wide range of λ_0 among fluorides, chlorides, and bromides with values up to 8 μm. The effect of a change in the halogen can be seen in the sequences of fluoride, chloride, bromide, and iodide for Li (λ_0 = 1.3, 2.3, 2.7, and 3.4 μm) and for divalent Pb (λ_0 = 3.3, 5.1, 7.4, and 13.9 μm).

Using eqs. (4) to calculate λ_0 for mixtures should be an improvement over assuming linearity between values of λ_0 for the end members in binary mixtures, and can also be used for more complex compositions. Some sample calculations, e.g., for $LiTlCl_2$, $SiGeO_4$, Mg_2OCl_2, etc., show that the two approaches range from agreement to producing a 15% difference, in either direction.

Keeping in mind the limitations discussed in the appendix, Figs. 1 and 2 indicate a number of compositions worthy of investigation. Note that Goodman[1] limited his thinking to wavelengths of $3\frac{1}{2}$ to 4 μm because this is an atmospheric transmission window and it might represent the limit of room temperature sources and detectors. The first constraint is not relevant to a normal fiber system containing no atmospheric linkages and the latter could also be unimportant if, say, a sufficiently low loss permitted a trans-Atlantic link without repeaters: Operation of the two cooled terminals would seem to be a reasonable price for such a system. In the oxides of Fig. 1, the currently used compositions based on SiO_2 and involving relatively small amounts of other oxides such as GeO_2 provide a λ_0 near 1.3 μm, conveniently centered in a "water" window of SiO_2. Use of the next window at about 1.6 μm could be possible in SiO_2 with a larger addition of GeO_2, with smaller amounts of many other possible oxides, or in a single mode fiber at zero total dispersion by balancing the waveguide and materials dispersions.[16] The limiting scattering loss for pure GeO_2 at its λ_0 of about 1.7 μm has been estimated at 0.15 dB/km.[17] Note that widely scattered λ_0 values for the SiO_2-GeO_2 system have been reported, apparently none above 16 percent GeO_2, however.[18]

Shifting our focus to wavelengths above 2 μm necessitates omitting the light Si atom. One possibility now centers about compositions containing the strong glass former GeO_2; combinations of this with

many of the weak or conditional glass formers in Fig. 1, such as PbO,[19] Bi_2O_3,[20] and PbO plus In_2O_3,[21] are known to give glasses. The other possibility is to use mixtures containing only the weak or conditional glass formers by themselves. The scope for a λ_0 above 3 μm appears to be quite restricted, with the availability of only Bi_2O_3, PbO, and the very toxic Tl_2O, all with relatively small values of E.

The highest value of λ_0 among the fluorides of Table II and Fig. 2 is the 3.5 μm of TlF. The glass-forming fluorides have relatively low λ_0 values, varying from the 1.2 μm of BeF_2 and AlF_3 to the 1.7 to 1.8 μm of ZrF_4 and HfF_4. Fluoride glasses based on some of these compounds have recently been investigated for waveguide use in the infrared. Calculations, however show relatively low λ_0 values of 1.91, 1.81, and 1.79 μm for three typical compositions: 57.5 HfF_4, 33.75 BaF_2, 8.75 LaF_2[5]; 40 BaF_2, 60 ZrF_4, and 10 GdF_3, 60 ZrF_4, 30 BaF_2,[6] respectively. Thus λ_0 values of fluoride glasses will probably not be significantly larger than the 1.8 μm of GeO_2 with its much lower reactivity and simpler chemistry. In the absence of the discovery of new glass-forming fluoride systems, prospects for fluoride glasses with λ_0 in the region beyond 2 μm are therefore not very promising.

In the case of chlorides, bromides, and iodides there is only one known glass former, $ZnCl_2$. Some $ZnCl_2$ fiber has been prepared,[4] but because of its hygroscopicity, only very limited studies have been performed so far. Weak glass formers leading to mixed halide glasses may exist here, as yet undiscovered. The existence of glass formation in mixed oxyhalides based on a combination of Figs. 1 and 2 is another possibility by analogy with lead oxyfluoride glass.[22]

V. MATERIALS PROPERTIES

Given the above conditions, a potentially useful glass for a low-loss infrared optical waveguide would need to be readily melted and worked, stable to devitrification and phase separation, capable of being adequately purified at a not unreasonable cost, and have the necessary mechanical and chemical properties, particularly adequate strength and low hygroscopicity. The ability to modify the refractive index without large changes in the thermal expansion is also needed for index profile adjustment to obtain minimum mode dispersion in multimode fibers.

In view of these considerations, Tables I and II have been generally limited to those oxides, fluorides, chlorides, and bromides with $E > 4\frac{1}{2}$ which can be melted without excessive decomposition. Fluorides seem to have λ_0 values that are too low and tend to be highly reactive and iodides are too unstable to be serious contenders, except possibly in small amounts for refractive index adjustment. A large hygroscopicity, as in $ZnCl_2$, can be a barrier, but such a substance may still be a useful

component in a mixed composition. The high toxicity of some elements, such as Tl, As, and some fluorides, are strong minuses. Chalcogenides and pnictides tend to be soft, brittle, somewhat reactive during preparation, and have low E and bandgap values.

VI. SUMMARY

We have calculated the wavelength for the material dispersion zero, i.e., that wavelength for which a multimode lightguide would function at highest bandwidth, for oxides and halides using fundamental properties. The results, as presented in the tables and figures, can be used to direct the search for long-wave lightguides, subject to a number of materials considerations.

VII. ACKNOWLEDGMENTS

It is a pleasure to thank S. Wemple for extended discussions and for his cooperation in estimating E and f values, and D. L. Wood for helpful comments on the manuscript.

APPENDIX

Material Constraints on Ultralow-Loss Infrared Optical Waveguides

A1. Types of fibers

Many types of optical waveguide fibers have been proposed, including liquid filled ones.[23] Glass has demonstrated its utility, but the use of single crystals and even of polycrystalline materials and plastically deformable Ag and Tl halides has also been proposed. In view of the basic stepped nature of single crystal surfaces and of the inherent scattering at grain and subgrain boundaries in the other nonvitreous materials, the applicability of material other than glass to ultralow-loss fibers remains highly questionable.

A2. Bulk losses

Four fundamental types of losses are important[24]:

(*i*) the electronic transition ultraviolet absorption tail loss (Urbach behavior, main optical absorption edge);

(*ii*) the scattering loss, derived from both intrinsic and extrinsic inhomogeneities, dominant on the UV side caused by the λ^{-4} Rayleigh variation (Raman and Brillouin effects are not important[24]);

(*iii*) the impurity losses, predominantly from "water" and 3d transition metal impurities, although rare earth metals may be more important in the infrared[1]; and

(*iv*) the vibrational states IR absorption tail losses (reststrahlen, highest energy optical phonon).

Both the UV and the IR loss regions show an exponential drop-off

with energy, thus creating a "window," with minimum loss at the crossover point (in the absence of impurity losses). Values as low as 10^{-6} dB/km are indicated[1] by extrapolations (which are not necessarily valid). Ideally, this minimum-loss wavelength would also be close to the location of λ_0 for an ultralow-loss long-distance fiber link.

A number of general principles for minimizing the various losses can be listed. For minimum UV tail and scattering losses, a large bandgap in a material with a low glass-transition temperature, a one or few component system, low isothermal compressibility, an open structure with a low density of anions and a low formal valence on the anion, and low coordination with a short nearest-neighbor bond length might be best.[4,25] For a minimum IR tail, a large bandgap, heavy atoms with weak (covalent) bonding, and low melting and glass-transition temperatures would lead to lower frequency vibrational states; most important is a very low anharmonicity so as to minimize overtones. Low impurity levels and a good homogeneity are obvious. Inevitably some of these preferences are mutually incompatible, particularly the high bandgap favoring both low UV and IR tails, as against a low E (a function of the bandgap) lowering λ_0 in eqs. (1) and (2).

REFERENCES

1. C. H. L. Goodman, Solid State and Electron Devices, *2* (1978), p. 129.
2. D. A. Pinnow, A. L. Gentile, A. G. Standlee, and A. S. Timper, Appl. Phys. Lett., *33* (1978), p. 28.
3. J. H. Garfunkel, R. A. Skogman, and R. A. Walterson, Proceedings IEEE/OSA Conference on Laser Engineering and Applications (1979), p. 49.
4. L. G. Van Uitert and S. H. Wemple, Appl. Phys. Lett., *33* (1978), p. 57.
5. M. G. Drexhage, C. T. Moynihan, and M. Saleh, Mater. Res. Bull., *15* (1980), p. 213.
6. S. Takahashi, S. Shibata, T. Kanamori, S. Mitachi, and T. Manabe, Am. Ceram. Soc. Abstr., April 1980.
7. M. Poulain, J. Lucas, and P. Brun, Mat. Res. Bull., *10* (1975), p. 243.
8. A. Lecog and M. Poulain, J. Noncrystal. Solids, *34* (1979), p. 101.
9. S. H. Wemple, Appl. Opt., *18* (1979), p. 31.
10. S. H. Wemple, J. Chem. Phys., *67* (1977), p. 2151.
11. H. Rawson, *Inorganic Glass-Forming Systems*, New York: Academic, 1967, pp. 8ff, 245–8.
12. K. H. Sun, Glass Tech., *20* (1979), p. 36.
13. C. Herzberg, *Infrared and Raman Spectra of Polyatomic Molecules*, New York: Van Nostrand, 1945, pp. 58, 171, 272, and 280.
14. P. Kaiser, A. R. Tynes, H. W. Astle, A. D. Pearson, W. G. French, R. E. Jaeger, and A. H. Cherin, J. Opt. Soc. Am., *63* (1973), p. 1141.
15. R. J. Pressley, ed., *Handbook of Lasers*, Cleveland, Ohio: Chemical Rubber Co., 1971, pp. 53–62.
16. L. G. Cohen, C. Lin, and W. G. French, Electron. Lett., *15* (1979), p. 334.
17. R. Olshansky and G. W. Scherer, Optical Communications Conference, Amsterdam, Sept. 1979, Abstract 1.103.
18. N. J. Adams, D. N. Payne, F. M. E. Sladen, and A. H. Hartog, Electron. Lett., *14* (1978), p. 703.
19. B. Phillips and M. G. Scroger, J. Am. Chem. Soc., *48* (1965), p. 398.
20. W. H. Dumbaugh, Phys. Chem. Glasses, *19* (1978), p. 121.
21. M. K. Murthy, Phys. Chem. Glasses, *15* (1974), p. 32.
22. W. A. Weyl and E. C. Marboe, *The Constitution of Glasses*, New York: Interscience, 1964, Vol. 2, Part 1, p. 592ff.
23. W. G. French, R. E. Jaeger, J. B. MacChesney, S. R. Nagel, K. Nassau, and A. D.

Pearson in *Optical Fiber Telecommunications*, S. E. Miller and A. G. Chynoweth, Eds., New York: Academic, 1979, p. 233ff.
24. B. G. Bagley, C. R. Kurkjian, J. W. Mitchell, G. E. Peterson, and A. R. Tynes in *Optical Fiber Telecommunications*, S. E. Miller and A. G. Chynoweth, Eds., New York: Academic, 1979, p. 167ff.
25. D. A. Pinnow, T. C. Rich, F. W. Ostermayer, and M. DiDomenico, Appl. Phys. Lett., *22* (1973), p. 527.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 3, March 1981
Printed in U.S.A.

On Newton-Direction Algorithms and Diffeomorphisms*

By I. W. SANDBERG

(Manuscript received August 29, 1980)

This paper reports on results that complement those in an earlier paper by this writer which gives a constructive proof of the existence of an algorithm that, for each right-hand side a, produces a sequence which converges globally and superlinearly to a solution x of f(x) = a whenever f is a C^1-diffeomorphism (i.e., is a continuously-differentiable invertible map with continuously-differentiable inverse) of a Banach space B onto itself and either $B = R^n$ or f satisfies certain other conditions that are often met in applications. Here we consider the case in which f′ is Lipschitz on each bounded subset of B. We give results which, while along the lines of those obtained earlier, concern a fundamentally different Newton-direction algorithm which does not appear to have been introduced previously, and which has the advantage that its implementation does not require the use of certain search procedures.

I. INTRODUCTION

Let f be a function from U into B, where B is a Banach space with norm $|\cdot|$, and U is a nonempty open subset of B. We say that f is *differentiable* on a set $S \subset U$ if f has a Frechet derivative $f'(s)$ at each point s of S.† (If, for example, $B = R^n$ with the usual Euclidean norm, then f is differentiable on U if it is continuously differentiable on U in the usual sense.) By f a *C^1-diffeomorphism*, we mean that f is a homeomorphism of U onto B, and f' and $(f^{-1})'$ exist and are continuous on U and B, respectively. (We emphasize that here *continuity* refers to the dependence of the derivatives on the points at which they are

* This paper was presented at the Fourteenth Asilomar Conference on Circuits, Systems, and Computers (Pacific Grove, California, November 17–19, 1980).

† In other words, f is *differentiable* on $S \subset U$ if for each $s \in S$, there is a bounded linear map $f'(s): B \to B$ such that $f(s+h) = f(s) + f'(s)h + o(|h|)$ as $|h| \to 0$.

evaluated, not to their boundedness as operators, which is assured by definition*) C^1-diffeomorphisms frequently arise in applications.

The purpose of this paper is to report on results that complement those in Ref. 1 where a constructive proof is given of the existence of a Newton-direction algorithm that, for each $a \in B$, generates a sequence in U which converges globally and superlinearly to a solution x of $f(x) = a$ whenever f is a C^1-diffeomorphism of U onto B and either $B = R^n$ or f satisfies certain other conditions that are frequently met in applications. (For the case of an important class of *monotone* diffeomorphisms f in a Hilbert space H, the "other conditions" reduce to simply the requirement that f' be uniformly continuous on closed bounded subsets of H. A specific example in which H is infinite dimensional is given in Ref. 1.)

The algorithm described in Ref. 1 typically involves the recursive determination of positive scalars $\gamma_0, \gamma_1, \cdots$ (which determine the successive steplengths) such that a certain ratio $R_k(\gamma_k)$ (which depends on the kth iterate x^k) lies between prescribed bounds for all $k = 0, 1, 2, \cdots$. While it is proved that the γ_k can be chosen as required, and that $\gamma_k = 1$ for all sufficiently large k, the actual determination of the γ_k in a specific case would ordinarily require the use of a one-dimensional search procedure for a finite (and possibly large) number of values of k.

In this paper we address the case in which $U = B$ and f' is *Lipschitz* on bounded subsets of B (i.e., is such that for each bounded subset S of B there is a constant Λ such that $|f'(u) - f'(v)| \leq \Lambda |u - v|$ for all u and v in S). We give results which, while along the lines of those in Ref. 1, concern a fundamentally different Newton-direction algorithm that does not appear to have been introduced earlier, and which does not require the use of search procedures to solve subproblems of the type outlined above.

Our results are presented in Section II. As a consequence of the Lipschitz hypothesis, proofs are comparatively simple and we are able to establish quadratic (rather than superlinear) convergence. (Recall that a sequence $x^1, x^2, \cdots$ in B *converges quadratically* to an element x of B if the sequence converges to x and there is a constant c such that $|x^{k+1} - x| \leq c|x^k - x|^2$ for all k.†)

General relationships between diffeomorphisms and computation of the type described in Ref. 1 and in Section II do not appear to have

* And of course, this continuity is with respect to the usual induced norm of a bounded linear map of B into B.

† Quadratic convergence results follow easily from those in Ref. 1 under the Lipschitz hypothesis used here. (In this connection, see the last part of the proof of Lemma 1 in Section II.)

been reported on earlier by other writers. On the other hand, as in Ref. 1 our approach involves the minimization of a functional, and therefore in a general sense there is a vast related literature. [See, for example, Ref. 2 and note (p. 190) that the least-squares Newton-direction methods described there require, in particular, the existence of *second* derivatives of f (our notation).] Additional background material can be found in Ref. 1.

II. PROCESSES N_0 and N_1

Throughout this section we use the terms *Lipschitz* and *converges quadratically* in the way indicated in Section I, we denote the usual induced norm of a linear map A of B into B by $|A|$, and we take $U = B$.

With f differentiable on B, but not necessarily a C^1-diffeomorphism, and with x^0 and a any two elements of B, consider the following process, in which s_k denotes $|f(x^k) - a|$ whenever $x^k \in B$ is defined.

Process N_1: Choose $\rho \in [\frac{1}{2}, 1)$ and $\lambda > 0$. Do the following for $k = 0, 1, \cdots$

If $f(x^k) = a$, set $x^{k+1} = x^k$.

If $f(x^k) \neq a$, determine $\phi_k \in B$ such that

$f'(x^k)\phi_k = a - f(x^k)$. Then

1. Let $\gamma_k = (\lambda s_k)^{-1}$ if $s_k > 2\rho\lambda^{-1}$

$= 1$ if $s_k \leq 2\rho\lambda^{-1}$.

2. Let $y^{k+1} = x^k + \gamma_k\phi_k$.

3. Set $x^{k+1} = y^{k+1}$ if either $s_k > 2\rho\lambda^{-1}$ and $|f(y^{k+1}) - a| \leq [1 - (2\lambda s_k)^{-1}]s_k$, or $s_k \leq 2\rho\lambda^{-1}$ and $|f(y^{k+1}) - a| \leq \frac{1}{2}\lambda s_k^2$. If neither pair of conditions is met, replace λ by 2λ in Step 1 and the sentence preceding this sentence, and return to Step 1.

Our main result is the following.

Theorem 1: Suppose that f is a C^1-diffeomorphism of B onto B. Let f' be Lipschitz on bounded subsets of B, and let $|(f^{-1})'|$ be bounded on bounded subsets of B. Then for each a and each x^0, Process N_1 can be carried out, and $x^1, x^2, \cdots$ converges quadratically to the unique solution x of $f(x) = a$.

2.1 Proof of Theorem 1

Let a and x^0 be given.

We first prove two lemmas which concern cases in which f need not be a C^1-diffeomorphism. Let $L = \{v \in B: |f(v) - a| \leq |f(x^0) - a|\}$,

and let $\bar{L}$ denote $\{w + \alpha f'(w)^{-1}[a - f(w)]: w \in L, \alpha \in [0, 1]\}$ when $f'(\cdot)^{-1}$ exists on L. (Assuming that $\bar{L}$ is defined, notice that it is bounded if L is bounded and $|f'(\cdot)^{-1}|$ is bounded on L. This observation is used later in the proof of Theorem 1 and in connection with Lemmas 1 and 2, below.) With η a positive constant, consider the following process.

Process N_0: Choose $\rho \in [\frac{1}{2}, 1)$. Do the following for $k = 0, 1, \cdots$.

If $f(x^k) = a$, set $x^{k+1} = x^k$.

If $f(x^k) \neq a$, determine $\phi_k \in B$ such that

$f'(x^k)\phi_k = a - f(x^k)$. Then let

$$\gamma_k = (\eta s_k)^{-1} \quad \text{if} \quad s_k > 2\rho\eta^{-1}$$
$$= 1 \quad \text{if} \quad s_k \leq 2\rho\eta^{-1}.$$

Set $x^{k+1} = x^k + \gamma_k\phi_k$.

Lemma 1: Assume that L is bounded and that $f'(\cdot)$ and $f'(\cdot)^{-1}$ exist on L with $|f'(w)^{-1}| \leq K$ for $w \in L$ and some constant K. Assume also that $f'(\cdot)$ exists and is Lipschitz, with Lipschitz constant Λ, on $\bar{L}$. Then for $\eta \geq \Lambda K^2$,

(a) Process N_0 can be carried out, and for each k such that $s_k \neq 0$ we have

$$s_{k+1} \leq [1 - (2\eta s_k)^{-1}]s_k \quad \text{if} \quad s_k > 2\rho\eta^{-1}, \tag{1}$$

$$s_{k+1} \leq \tfrac{1}{2}\eta s_k^2 \leq \rho s_k \quad \text{if} \quad s_k \leq 2\rho\eta^{-1}. \tag{2}$$

(b) $s_k \to 0$ as $k \to \infty$.

(c) If there is an $x \in B$ such that $f(x) = a$ and $x^k \to x$ as $k \to \infty$, then $\{x^k\}$ converges quadratically.

2.1.1 Proof of Lemma 1

We will use the following proposition.

Proposition 1: Suppose that the hypotheses of Lemma 1 are met. If $x^k \in L$, and $\gamma \in [0, 1]$, and ϕ_k denotes $f'(x^k)^{-1}[a - f(x^k)]$, then, for $\eta \geq \Lambda K^2$, we have $|f(x^k + \gamma\phi_k) - a| \leq (1 - \gamma)|f(x^k) - a| + \frac{1}{2}\eta\gamma^2|f(x^k) - a|^2$.

Proof: We have

$$|f(x^k + \gamma\phi_k) - a| = |f(x^k) - a + f'(x^k)\gamma\phi_k + \delta|$$

in which

$$\delta = f(x^k + \gamma\phi_k) - f(x^k) - f'(x^k)\gamma\phi_k.$$

Thus

$$|f(x^k + \gamma\phi_k) - a| \leq (1 - \gamma)|f(x^k) - a| + |\delta|,$$

and

$$\delta = \int_0^1 [f'(x^k + \beta\gamma\phi_k) - f'(x^k)]\, d\beta \cdot \gamma\phi_k.$$

Since $|\delta| \leq \frac{1}{2}\Lambda\gamma^2 K^2 |f(x^k) - a|^2$, we have proved the proposition.*

Assume now that $\eta \geq \Lambda K^2$, and that the hypotheses of Lemma 1 are met.

Let k be such that either $k = 0$, or Process N_0 can be used to generate $x^1, \cdots, x^k$ with $x^j \in L$ for $j = 1, 2, \cdots, k$. Suppose that $s_k \neq 0$. Since $x^k \in L$, ϕ_k can be determined. Since $\rho \in [\frac{1}{2}, 1)$, when $s_k > 2\rho\eta^{-1}$ we have $(\eta s_k)^{-1} < 1$. Thus, by the proposition, (1) holds. On the other hand, obviously $\frac{1}{2}\eta s_k \leq \rho$ when $s_k \leq 2\rho\eta^{-1}$, and thus, by the proposition, (2) is met. This shows that x^{k+1} can be determined, that it satisfies (1) and (2), and that $x^{k+1} \in L$, which proves Part (a).

Part (b) is a direct consequence of Part (a), because, by Part (a), if s_k does not approach zero as $k \to \infty$ we must have $s_k > 2\rho\eta^{-1}$ for all k in which case $[1 - (2\eta s_0)^{-1}] \in (0, 1)$ and $s_k \leq [1 - (2\eta s_0)^{-1}]^k s_0$ for $k \geq 1$, which is a contradiction.

Assume now that B contains an x such that $f(x) = a$ and $x^k \to x$ as $k \to \infty$.† Since $x^k \in L$ for all k, and L is closed, $x \in L$. Let J denote $f'(x)$. Since J is an invertible bounded linear map of B into itself, there are positive constants β_1 and β_2 such that $\beta_1|u| \leq |Ju| \leq \beta_2|u|$ for $u \in B$. For each k, we have

$$|f(x^k) - a| = |f(x) - a + J(x^k - x) + \delta_k|$$

in which $|\delta_k|\,(|x^k - x|)^{-1} \to 0$ as $k \to \infty$.

Notice that for some m,

$$|J(x^k - x)| \geq 2|\delta_k| \qquad \text{for} \qquad k \geq m.$$

Thus for $k \geq m$,

$$|f(x^k) - a| \geq |J(x^k - x)| - |\delta_k| \geq \tfrac{1}{2}|J(x^k - x)| \geq \tfrac{1}{2}\beta_1|x^k - x|,$$

and, on the other hand,

$$|f(x^k) - a| \leq |J(x^k - x)| + |\delta_k| \leq \tfrac{3}{2}|J(x^k - x)| \leq \tfrac{3}{2}\beta_2|x^k - x|.$$

We have $s_{k+1} \leq \frac{1}{2}\eta s_k^2$ for $k \geq M$ for some $M \geq m$.

---

* With regard to the origin of the formula for γ_k in Process N_0, notice that the right side of the main inequality of the proposition is minimized with respect to γ at $\gamma = (\eta s_k)^{-1}$.

† The existence of such an x follows from our hypotheses, but this fact is not needed for our purposes.

Therefore,

$$|x^{k+1} - x| \leq \tfrac{3}{4}\,\eta\beta_1^{-1}\,\beta_2^2\,|x^k - x|^2, \qquad k \geq M,$$

which completes the proof of the lemma.*,†

Lemma 2: Suppose that L is bounded, and that $f'(\cdot)$ and $f'(\cdot)^{-1}$ exist on L with $|f'(\cdot)^{-1}|$ bounded on L. Suppose also that $f'(\cdot)$ exists and is Lipschitz on $\bar{L}$. Then Process N_1 can be carried out, we have $s_k \to 0$ as $k \to \infty$, and if there is an $x \in B$ such that $f(x) = a$ and $x^k \to x$ as $k \to \infty$, then $x^1, x^2, \cdots$ converges quadratically.

2.1.2 Proof of Lemma 2

Consider Process N_1. By Lemma 1, there is a constant λ_0 that depends only on f, a, and x^0 such that if λ in Step 1 and the first sentence of Step 3 satisfies $\lambda \geq \lambda_0$, and if either $k = 0$ and $s_0 \neq 0$, or $k > 0$ and Process N_1 can be used to determine x^k with $s_k \neq 0$ and $s_k \leq s_0$, then Step 3 can be carried out on the first pass. Notice that whenever x^{k+1} is set equal to y^{k+1} in Step 3, we have $s_{k+1} < s_k$.

Since for any $\lambda > 0$ there is a nonnegative integer p such that $2^p\lambda \geq \lambda_0$, it follows that Process N_1 can be carried out, and that for some nonnegative integers q and r, we have $s^{k+1} \leq \sigma_k s^k$ for $k \geq q$, where $\sigma_k = [1 - (2^{r+1}\lambda s_q)^{-1}]$ when $s^k > 2\rho(2^r\lambda)^{-1}$ and $\sigma_k = \rho$ otherwise. Since $\sigma_k < 1$ for $k \geq q$, it is clear that $s_k \to 0$ as $k \to \infty$, and therefore that $s_{k+1} \leq \tfrac{1}{2}\,2^r\lambda s_k^2$ for $k \geq M$ for some M. Thus, by the proof of Part (c) of Lemma 1, our proof of Lemma 2 is complete.

Now let the hypotheses of Theorem 1 be met. The proof of Theorem 3 of Ref. 1 shows that L is bounded, that $f'(\cdot)^{-1}$ exists on B, and that $|f'(\cdot)^{-1}|$ is bounded on L. Since f is a homeomorphism of B onto B, $s_k \to 0$ as $k \to \infty$ implies that $x^k \to x$ as $k \to \infty$, where x satisfies $f(x) = a$. By Lemma 2, this completes the proof of Theorem 1.

2.2 Monotone diffeomorphisms in Hilbert space

Let $\psi:[0, \infty) \to [0, \infty)$ be continuous, strictly increasing, and such that $\psi(0) = 0$, $\psi(\alpha) \to \infty$ as $\alpha \to \infty$, and $\alpha^{-1}\psi(\alpha) \geq c$ for $\alpha \in (0, \bar{\alpha})$ for some positive constants c and $\bar{\alpha}$. Notice that, for example, $\psi(\alpha) = \alpha$ meets these conditions.

---

* The fact that $\{x^k\}$ converges quadratically follows from a direct extension of a known result (see Ref. 3, p. 312) since either $s_k = 0$ for some k, or there is an M such that $\gamma_k = 1$ for $k \geq M$. The short proof given above is included for the sake of completeness.

† D. J. Rose has informed this writer that in recent independent joint work with R. Bank,[4] done subsequent to the appearance of preprints of Ref. 1, a corresponding result, as well as a result corresponding to Theorem 1, was obtained for a process in which $\gamma_k = (1 + \eta_k s_k)^{-1}$, where the η_k satisfy certain inequalities. They study a case in which an approximation M_k to $f'(x^k)$ can be used in place of $f'(x^k)$. Also, earlier related work along different lines concerning uniformly monotone gradient maps $f:R^n \to R^n$ was done by Bank and Rose.[5]

Theorem 2: Let f map a real Hilbert space H, with inner product $\langle \cdot, \cdot \rangle$, into itself such that $\langle f(u) - f(v), u - v \rangle \geq |u - v| \psi(|u - v|)$ for all $u, v \in H$. Assume that f' exists and is Lipschitz on bounded subsets of H. Then f is a C^1-diffeomorphism of H onto H, and the conclusion of Theorem 1 holds.

Using Theorem 1, a proof of Theorem 2 can be obtained by trivially modifying the proof of Theorem 4 of Ref. 1.

2.3 $B = R^n$

The following complete result is a direct corollary of Theorem 1 (see the proof of Theorem 5 of Ref 1).

Theorem 3: Let $B = R^n$, and let f' be Lipschitz and continuously differentiable on bounded subsets of R^n. Then f is a C^1-diffeomorphism of R^n onto itself if and only if

(i) Process N_1 can be carried out for each a and each x^0.

(ii) For each a, the sequence produced by Process N_1 converges quadratically to a solution x of $f(x) = a$, and x does not depend on x^0.

2.4 Comments

As in Ref 1, our primary purpose is to focus attention on general relationships between diffeomorphisms and computation. Clearly, no attempt is made to optimize the performance of all aspects of the type of algorithm described. However, there are some basically self-evident modifications that are sometimes useful. For example, the total number of iterations required in a specific case can sometimes be reduced significantly by repeatedly, or occasionally, stopping the algorithm after a number of steps and resetting the initial value of λ in Process N_1 to a smaller number. (It is not difficult to give rules of thumb concerning *when* to stop the algorithm and by *how much* to reduce λ, but we have not tried to prove theorems that bear on these matters.) Of course, bounds on the location of the solution and estimates of K and Λ, which are available in some problems, can be used in an obvious way. Similarly, if for example $B = R^n$, a globally convergent steepest-descent process (see Ref. 6)* might be used initially to obtain a better approximation to the solution before the Newton-direction algorithm is used. (In fact, a well known and often useful strategy is to combine steepest descent and *pure* Newton iterations in this way.†)

---

* We take this opportunity to correct a typographical error in Ref. 6. On page 1004, left column, line 2, [2] should be replaced with [21].

† This paragraph was motivated by a helpful observation by D. J. Rose to the effect that, as the algorithm stands, there are cases in which many iterations are required.

REFERENCES

1. I. W. Sandberg, "Diffeomorphisms and Newton-Direction Algorithms," B.S.T.J., *59* (November 1980), pp. 1721–34.
2. J. W. Daniel, *The Approximate Minimization of Functionals*, Englewood Cliffs, N.J.: Prentice-Hall, 1971.
3. J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, New York: Academic, 1970.
4. D. J. Rose and R. E. Bank, "Global Approximate Newton Methods," to appear.
5. R. E. Bank and D. J. Rose, "Solving Nonlinear Systems of Equations Arising from Semiconductor Device Modelling," unpublished work, January 1980.
6. I. W. Sandberg, "Global Inverse Function Theorems," IEEE Trans. Circuits and Systems, *CAS-27*, No. 11 (November 1980), Special Issue on Nonlinear Circuits and Systems, pp. 998–1004.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 3, March 1981
Printed in U.S.A.

Program Development by Stepwise Refinement and Related Topics

By N. GEHANI

(Manuscript received September 22, 1980)

Computer program development by stepwise refinement has been advocated by many people. We take another look at stepwise refinement in light of recent developments in programming languages and programming methodology such as abstract data types, correctness proofs and formal specifications, parallel programs and multiversion programs. We offer suggestions for the refinement process and discuss program maintainability.

I. INTRODUCTION

The correct design of nontrivial programs and systems of programs is an intellectually challenging and difficult task. Often programs are designed with very little time spent on the design itself, the effort being concentrated on coding. This could be due to management's desire to see something working as soon as possible to be assured that work is progressing, or it could be due to the programmer's desire to "attack the problem right away."

Not only is there no emphasis on design, the approach to it is also not systematic or disciplined. This results in programs that do not meet specifications in terms of correct output and performance requirements.

What we want is a programming methodology that puts some discipline and structure in the design process without stifling creativity. A programming methodology should:

(*i*) Help us master the complexity of the problem being solved and give us some guidelines on how to formulate the problem solution.

(*ii*) Provide us with a written record of the design process. The design can then be read by others, and the design decisions can be appreciated or constructively criticized.

(*iii*) Result in programs that are understandable.

(*iv*) Lead to programs whose correctness can be verified by proofs. Since proofs are difficult, the methodology should allow for a systematic approach to program testing.

(*v*) Be generally applicable and not restricted to a class of problems.

(*vi*) Allow for the production of efficient programs.

(*vii*) Allow for the production of programs that can be modified systematically.

In this tutorial we discuss a programming methodology called stepwise refinement and informally show that it satisfies these criteria.

II. STEPWISE REFINEMENT

Stepwise refinement is a top-down design approach to program development (first advocated by Wirth[4]). Wirth really gave a systematic formulation and description of what many programmers were previously doing intuitively. According to Brooks,[2] stepwise refinement is the most important new programming formalization of the decade. Stepwise refinement is applicable not only to program design, but also to the design of complex systems.

In a top-down approach, the problem to be solved is decomposed or refined into subproblems which are then solved. The decomposition or refinement should be such that:[3,4]

(*i*) The subproblems should be solvable.

(*ii*) A subproblem should be solvable with as little impact on the other subproblems as possible.

(*iii*) The solution of each subproblem should involve less effort than the original problem.

(*iv*) Once the subproblems are solved, the solution of the problem should not require much additional effort.

This process is repeated on the subproblems; of course, if the solution of a problem is obvious or trivial, then this decomposition is not necessary.

If P_0 is the initial problem formulation/solution, then the final problem formulation/solution P_n (an executable program) is arrived at after a series of gradual "refinement" steps,

$$P_0 \Rightarrow P_1 \Rightarrow P_2 \Rightarrow \cdots \Rightarrow P_n.$$

The refinement P_{i+1} of P_i is produced by supplying more details for the problem formulation/solution P_i. The refinements $P_0, \cdots, P_n$ represent different levels of abstraction. P_0 may be said to give the most abstract view of the problem solution P_n, while P_n represents a detailed version of the solution for P_0.

As an example of abstraction levels, consider a program that automates the record-keeping of an insurance company. At the highest level of abstraction, the program deals with the insurance company as

an entity. At succeeding lower levels of abstraction, the program deals with

- different insurance categories (auto, home, life, etc.)
- groups of policies in the above categories
- individual policies in the above groups
- details of individual policies

Each refinement P_i consists of a sequence of instructions and data descriptions P_{ij},

$$\begin{matrix} P_{i1} \\ \vdots \\ P_{in_i} \end{matrix}$$

In each refinement step, we provide more details on how each P_i is to be implemented. The refinement process stops when we reach a stage

(*i*) where all the instructions can be executed on a computer, or

(*ii*) where instructions can be easily translated to computer executable instructions.

Pictorially, the refinement process may be depicted as shown in Fig. 1. The final program is a collection of the nodes at the last refinement level P_n.

The design can be probed to any desired level of detail i ($0 \leq i \leq n$). Understanding the design process is aided by the fact that level i provides an overview of levels $i + 1$ through n.

We illustrate the stepwise refinement process with annotated examples. The notation we will use for conveying our ideas will be Pascal-like[5] and include guarded commands.[6] PL/I will be used to show the executable versions of some programs.

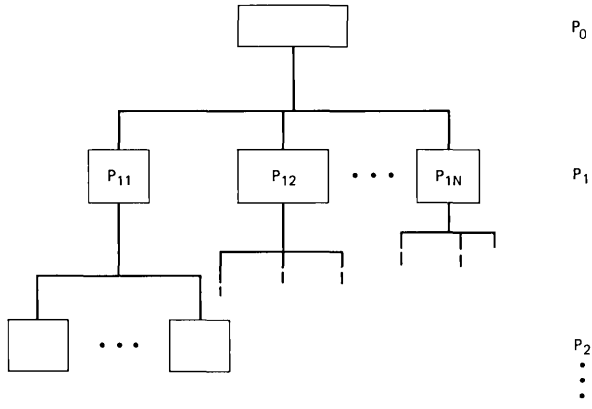


Fig. 1—The refinement process.

The guarded commands are

(*i*) *Selection*

```
if  b1   →   SL1
 []b2    →   SL2
          ⋮
 []bn    →   SLn
fi
```

The b_i's are called the guards (Boolean expressions) and the SL_i are statement lists. For a successful execution of the selection statement, at least one of the guards must be true. If only one guard is true, then the corresponding statement list is executed. If more than one guard is true, then one of the corresponding statement lists is selected nondeterministically (i.e., the user cannot tell beforehand) and executed, e.g.,

```
if  a ≥ b   →   max := a
 []b ≥ a    →   max := b
fi
```

If $a = b$, then both the guards are true and either of the statements max := a or max := b may be executed. Either way, the answer is right. This symmetry is aesthetically pleasing when compared to conventional deterministic programming.

(*ii*) *Repetition*

```
do  b1   →   SL1
  []b2   →   SL2
             ⋮
  []bn   →   SLn
od
```

The loop is repeatedly executed as long as one of the guards is true. If one guard is true, then the corresponding statement list is executed. As in the selection statement, if more than one guard is true, then one of the corresponding lists is arbitrarily selected and executed.

Implementation of these statements in *C*, Pascal, PL/I, etc., will be deterministic. For example, in PL/I:

(*i*) *Selection*

```
IF b1
  THEN DO; SL1; END;
  ELSE IF b2 THEN DO; SL2; END;
       ⋮
       ELSE IF bn THEN DO; SLn; END;
            ELSE ERROR;
```

(*ii*) *Repetition*

```
L:DO WHILE ('1' B);
        IF b1
           THEN DO; SL1; END;
           ELSE IF b2 THEN DO; SL2; END;
                   ⋮
                ELSE IF bn THEN DO; SLn; END;
                     ELSE GOTO LE;
        END L;
LE:;
```

Note: These statements could be more conveniently implemented using the new PL/I SELECT and LEAVE statements.

III. EXAMPLES OF STEPWISE REFINEMENT

The examples used to illustrate stepwise refinement are small out of necessity. The reader is encouraged to apply stepwise refinement to larger problems.

Example 1

Write a program to simulate a week in John's life.

Initial refinement P_0:

Simulate a week in John's life

If we were programming in a language that understood the above instruction, then we wouldn't have to refine it further.

Refinement P_1:

```
a.    d := monday {next day to be simulated is d}
b.    repeat
c.       simulate day d in John's life
d.       d := next day
e.       until week over
```

A refinement consists of programming language instructions mixed with English statements.

Refinement P_2:

Line c of P_1 is refined as

```
Sleep until alarm goes off
Go through morning ritual
Spend the day
Go through evening ritual
Prepare to sleep
```

Line d is refined as

```
if  d = sunday   →   d := monday
 []d ≠ sunday    →   d := SUCC(d)
fi
```

where the Pascal function SUCC gives the next day in the range of values monday, tuesday, $\cdots$, sunday. Line e: "week over" is refined as "d = monday."

Collecting these refinements of P_1's instructions, we get refinement P_2.

```
d := monday
repeat
    Sleep until alarm goes off
    Go through morning ritual
    Spend the day
    Go through the evening ritual
    Prepare to sleep
    if  d = sunday   →   d := monday
     []d ≠ sunday    →   d := SUCC(d)
    fi
    until d = monday.
```

This collection can be done mechanically and we shall in general omit it.

"Spend the day" may be refined as

```
if  weekday   →   go to work
                  work
                  return home
 []weekend    →   read newspaper
                  laze around
                  read book
                  watch TV
fi
```

Similarly, the other instructions of P_2 may be refined and the refinement process continued to the desired level of detail. In the refinement we have tried to model processes of the problem domain.[7]

An initial decomposition might not be feasible or nice, in which case we back up and try another decomposition. We shall only present the final set of decompositions.

Example 2

Write a program that reads in a list of positive numbers a_1, a_2, $\cdots$, a_n ($n \geq 0$) and prints the sums of all natural numbers up to each a_i, i.e., the sums:

$$\sum_{i=0}^{a_1} i, \quad \sum_{i=0}^{a_2} i, \quad \cdots, \quad \sum_{i=0}^{a_n} i.$$

Initial refinement P_0: Print $\sum_{i=0}^{a_1} i, \sum_{i=0}^{a_2} i, \cdots, \sum_{i=0}^{a_n} i$.

P_1^*:

```
read a
do while there
   exists data   →   Compute sum = Σ_{i=0}^{a} i.
                     Print sum
                     read a
od
```

Because we are aiming for an executable program in a sequential programming language, the refinement P_1 reflects the decision to read in an input element, compute its sum, print the sum, and then read another input element. Alternately, had our target been a parallel computer we would have probably read in all the input elements, computed the sums in parallel, and then printed them out. *Many implicit decisions underlie every refinement.*

P_2:
- while there exist data
  is refined to
  not EOF
- Compute sum = $\sum_{i=0}^{a} i$
  is refined to

```
i := 0
sum := 0          {sum = 0 + 1 + 2 + ··· + i}
do i ≠ a    →     i := i + 1; sum := sum + i od
```

Let us now examine the concept of a loop invariant. A loop invariant is an assertion about program variables; it statically captures the meaning of a loop thus helping us understand it. Loop invariants are true before and after the execution of a loop, and before and after each execution of the loop body. Dijkstra[6] suggests some ways of finding the loop invariant using the desired post-condition (state of variables after the loop terminates). The loop invariant can actually aid in determining the guards and the corresponding statement lists.

Let I be the loop invariant sum = $0 + 1 + 2 + \cdots + i$. I is true initially because $i = 0$ and sum = 0. Evaluation of the guard $i \neq a$ does not affect I; the statement $i := i + 1$ destroys I, resulting in sum = $0 + 1 + 2 + \cdots + i - 1$. But sum = sum + i restores the validity of invariant I. When the guard evaluates to false, i.e., $i = a$, the loop terminates. Now in addition to I being true we have $i = a$, implying the desired result sum = $0 + 1 + 2 + \cdots + a$.

How can we demonstrate loop termination? For this we must show the existence of a function, initially ≥ 0, whose value is decreased by

---

* The fact that we have two read statements in P_1 shows that our design is influenced by our target language (PL/I in this case). In PL/I, unlike in Pascal, a read must occur on an empty file before an end-of-file is indicated (via the variable EOF in our case).

one every time the loop is executed. When this function becomes ≤ 0, we stop. Such a function is $a - i$; executing $i := i + 1$ decreases its value by 1. When $a - i = 0$, we have $i = a$ which is when the guard evaluates to false and the loop terminates.

Continuing the refinements we get

P_3:

```
read a
do not EOF  →  { compute sum = Σ_{i=0}^{a} i }
               i := 0
               sum := 0                {sum = 0 + 1 + 2 + ··· + i}
               do i ≠ a  →  i := i + 1
                            sum := sum + i
               od
               print sum
               read a
od
```

P_4 (in PL/I):

```
SUM: PROC OPTIONS(MAIN);
 DCL (A /*NEXT INPUT ELEMENT */
      ,I /*LOOP VARIABLE */
      ,SUM /*SUM=0+1+2+···+I */
      )FIXED DEC,
      EOF BIT(1) INIT('O'B);
 ON ENDFILE EOF='1'B;

 GET LIST(A);
 DO WHILE (~EOF);
    I=0; SUM=0;
     DO WHILE (I~=A);
        I=I+1; SUM=SUM+I; END;
     PUT SKIP LIST (' SUM UPTO', A, ' IS ', SUM);
     GET LIST(A);
     END;
 END SUM;
```

Example 3

Write a program to determine the maximum element value of an $m \times n$ array A $(m, n \geq 1)$.

```
P0: determine the max element value of A
P1: i:=0    {last row examined}
    initialize max    {max is the maximum element
                       of rows 1 ··· i—loop invariant I1}
```

do all rows not examined → $i := i+1$; max := maximum(row i, max)
od

P_2:
- initialize max is refined to
  max := $A[1, 1]$
- all rows not examined
  is refined to
  $i \neq m$
- max := maximum (row i, max)
  is refined as
  $j = 0$ {max = maximum of rows $1 \ldots i-1$ and elements $1 \ldots j$ of row i—loop invariant I_2}
  do all elements of row i not examined → $j := j + 1$; max := MAX(max, $A[i, j]$)
  od

I_2 is the loop invariant. As an exercise, the reader should try and show that the loops leave I_1 and I_2 invariant, i.e., unchanged.

P_3:
- all elements of row j not examined
  is refined as
  $j \neq n$
- max := MAX(max, $A[i, j]$)
  is refined as

if max $\geq A[i, j]$ → skip
[]max $\leq A[i, j]$ → max := $A[i, j]$
fi

where skip denotes the null statement.

The iterative feature is the most important feature of a programming language.[8] The **do** ··· **od** construct allows us to express algorithms clearly and succinctly. The above example could have been done better had the author not used the **do** ··· **od** construct to just simulate the **while** statement. Making fuller use of the **do** ··· **od** construct, we get the following program for the above problem:

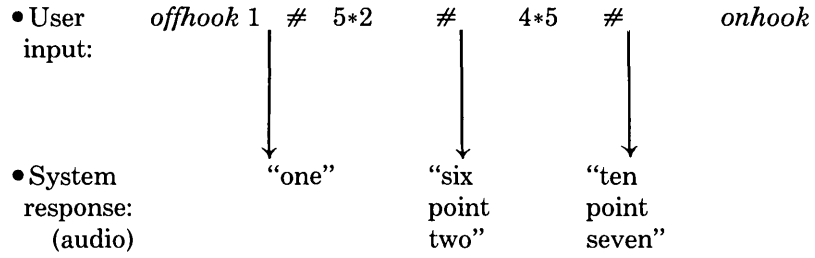
P_2': $i := 0$ {number of rows examined so far}
$j := 0$ {number of elements of row $i + 1$ examined so far}
initialize max {max is the maximum of all the elements in the first i rows and the first j elements of row $i + 1$} $- I$
do $i < m$ rows and $j < n$ elements of row $i + 1$ examined → $j := j + 1$; max := MAX(max, $A[i, j]$)
[]$i < m$ rows and all elements of row $i + 1$ examined → move to the next row
od

```
P'3:  i := 0;   j := 0
      max := A[1, 1]   {max is the maximum of all the
                        elements in the first i rows
                        and the first j elements of row i + 1} − I
      do  i < m and j < n   →   j := j + 1
                                max := MAX(max, A[i, j])
        []i < m and j = n   →   i := i + 1; j := 0
      od
```

Gries also shows that the **do** ··· **od** construct usually eliminates the need for loop exits necessary in programs that use the **while** statement.[8]

Example 4

The Touch-Tone® telephone provides an easy but limited means of communicating with a computer (see Fig. 2). The problem is to write a program that provides a simple adding machine to the user.[9] For example:



The characters # and * represent + and ·, respectively.

The following modules are available to the programmer:

• SPEAK (string)—provides an audio response for the number represented by the string.

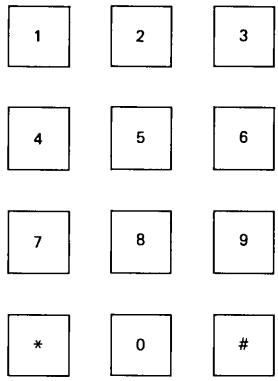
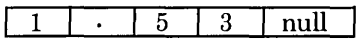
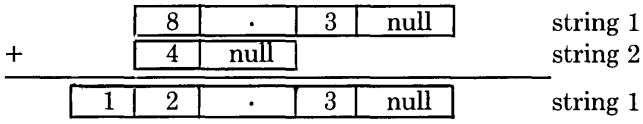


Fig. 2—The Touch-Tone® telephone's pushbutton dial.



- ADD(string1, string2) – string1 := string1 + string2



- waitsignal(char)—sets char to the next input character when available.

The input/output specifications written more formally are:

input: *offhook* f_1 # f_2 # ⋯ # f_n # *onhook*,

where $f_{i\ (1\le i\le n)}$ = { integer — one or more digits; real — one or more digits followed by one or more digits followed by * and at least one digit, or one * followed by at least one digit }

output: SPEAK(SUM$_1$), SPEAK(SUM$_2$), ⋯, SPEAK(SUM$_n$)
where SUM$_i = \sum_{k=1}^{i} f_k$, $1 \le i \le n$,
and the audio response occurs after the character # is input.

We assume that the maximum length of numbers input will be $k - 1$. To focus on the refinement process, we make the following additional assumptions:

(*i*) one addition session,
(*ii*) no errors of any kind.

In the second version of the solution we will eliminate these restrictions.

Refinement P_0: Do telephone addition.

```
P1:  Compute and speak out the running sum of the numbers
                          input.
P2:  plus := '#';  point := '*'
     waitsignal(c) {c contains the next
                    input char to be processed;
                    offhook is the first one}
     waitsignal(c) {get char after offhook}
     initially SUM is 0
     do c ≠ 'onhook'  →  read number into A
                         ADD(SUM, A)
                         SPEAK(SUM)
                         waitsignal(c) {+ consumed}
     od
```

The number variables SUM and *A* are implemented as strings because modules SPEAK and ADD expect strings as arguments.

Each variable is represented by

(*i*) variable of type string,

(*ii*) an integer variable that denotes the length of the string,

where **type** string = **array** [1 ··· *k*] **of** char.

In addition to SPEAK and ADD we need

(*i*) a procedure ZERO to initialize the numbers represented as strings to 0,

(*ii*) a procedure APPEND to help build a number.

They are defined as

```
procedure ZERO(var X:string; I:integer);
    begin I:=1; X[I]:=null end
procedure APPEND(var X:string; I:integer; c:char);
    begin X[I]:=c; I:=I + 1; X[I]:=null end
```

The implementation of numbers and their operations in terms of strings is an example of *data refinement.*

- Read number into *A* is refined as

```
ZERO A
do c ≠ plus  →  if c = point  →  APPEND '·' to A
                 [] c ≠ point  →  APPEND c to A
                fi
                waitsignal(c)
od
```

Collecting the refinements together we get

```
const point = '*'; plus = '#';
type string = array[1 .. k] of char;
var SUM, A:string;
    ISUM, IA:integer; {lengths of SUM, A}
begin waitsignal (c); waitsignal (c);
    ZERO(SUM, ISUM);
    do c ≠ 'onhook'  →  {read number into A}
            ZERO(A, IA)
            do c ≠ plus
               if c = point  →  APPEND(A, IA, '·')
               [] c ≠ point  →  APPEND(A, IA, c)
               fi
               waitsignal(c)
            od
            ADD(SUM, A);
            SPEAK(SUM);
            waitsignal(c)
```

```
od
end
```

Instead of including inline the refinement of "read number into A" in the final version of the program, it would have been more appropriate to make the refinement into a procedure READ and to call READ from the final version. This is because READ and the operations ADD and SPEAK (which appear in the final program) operate on numbers thus representing the same level of abstraction. Also, if an instruction appears more than once, then it should perhaps become a procedure call. The instruction would then be refined only once.

In the following version of the above program we eliminate the restrictions of a single session and no errors. The following types of errors are considered possible:

- illegal characters,
- $n \geq 2$ decimal points per number,
- only a decimal point—no digits,
- + follows +, *offhook*, i.e., null number,
- session starts with other than *offhook*.

The initial problem formulation P_0 is

do true → Do telephone addition **od**

P_1: • Do telephone addition is refined as
Compute and speak out the running sum of the numbers input so far.

This is refined as

P_2: waitsignal(c)

```
if c ≠ 'offhook'  →  error
[] c = 'offhook'  →  skip
fi
waitsignal(c)
check for valid char {digits, plus, point, 'onhook'}
initially sum is 0
 do c ≠ 'onhook'  →  Read number into A
                     if c = plus  →  ADD(SUM, A)
                                     SPEAK(SUM)
                                     waitsignal(c)
                                     check for valid char
                     [] c = 'onhook' → skip
                     fi
 od
```

Using the procedures APPEND and ZERO defined before, we refine read number into A as

```
ZERO A
# digits, # pts := 0;
do c is a
    digit or a point
                → if c is a point → Append '.' to A
                                    if # pts = 0 → # pts := # pts + 1
                                    [] # pts ≠ 0 → error
                                    fi
                  [] c is a digit → Append c to A
                                    # digits := # digits + 1
                  fi

    waitsignal(c);
    check for valid char
od
if # digits = 0 → error
[] # digits ≠ 0 → skip
fi
```

Continuation of this refinement process is similar to the error-free version and we omit it.

Example 5. McDonald's warehouse problem [9]

Given a list of item cards ordered by item number, produce the management report shown in Fig. 3. Each invoice has an item number, a code D for delivery, and R for received, and the quantity received or delivered.

P_0: Produce management report

P_1: a. Print heading
b. Process the item groups
c. Print number of item groups changed

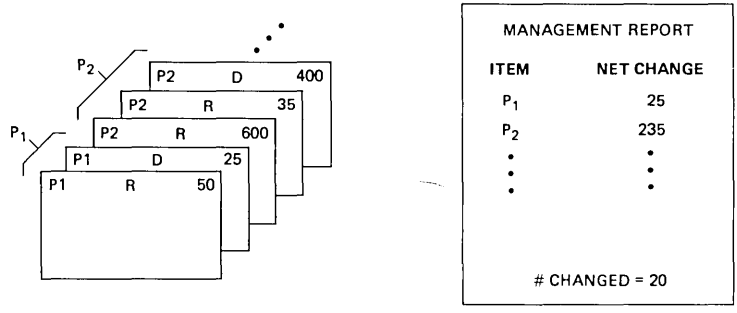


Fig. 3—From item cards to management report.

- Line b is refined as

```
# changed := 0
read item
do there are more → process an item group
   item groups      print item and net change
                    # changed := # changed + 1
od
```

- there are more item groups
  is refined as
  not EOF
- process an item group
  is refined as

```
netchange := 0
itemgroup # := item#
do item in group and not EOF
   → if code = R → netchange := netchange + Qty
     [] code = D → netchange := netchange − Qty
     fi
     read item
od
```

- item in group
  is refined as

itemgroup # = item#

This concludes the example.

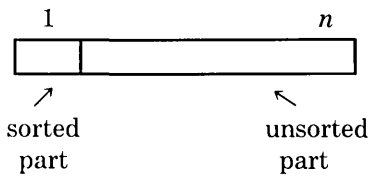
Example 6

Using insertion sort, sort the array A (size $n \geq 1$) in nondecreasing order, i.e., $A_1 \leq A_2 \leq \cdots \leq A_n$, and the new values of array A are a permutation of its old values.

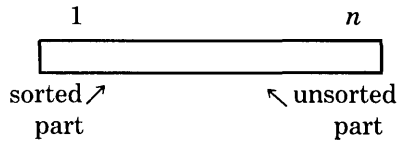
P_0: Sort the array A

Pictorially we can characterize the input and output specifications of the array A as

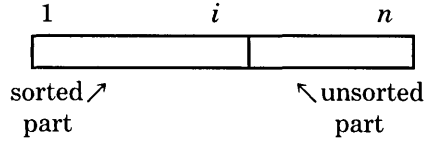
(i) initially



(*ii*) finally



At some intermediate stage of the sorting we will have



This picture corresponds to our loop invariant. It is true initially if $i = 1$. At the end of the loop, $i = n$ implies that the whole array is sorted. So the purpose of the loop body will be to exchange the values of A in such a way that i can be increased until it equals n. P_0 is therefore refined as

P_1:

$i := 1$ $\{A[1 \cdot\cdot i]$ sorted$\}$
do $i \neq n \rightarrow$ Extend sorted portion to include $A[i + 1]$
$i := i + 1$
od

- Extend sorted portion to include $A[i + 1]$ is refined as

  a. $t := A[i + 1]$
  b. shift all elements of $A[1 \cdots i] > t$ one place to the right such that $A[1 \cdots j - 1] \leq t$ and $A[j + 1 \cdots i + 1] > t$
  c. $A[j] := t$

- Line *b* of the above refinement is developed as

$j := i + 1$ $\{A[j + 1 \cdots i + 1] > t\}$
do $A[j - 1] > t \rightarrow$ shift $A[j - 1]$ to the right
$j := j - 1$
od

On loop termination we have $A[j + 1 \cdot\cdot i + 1] > t$ and $A[j - 1] \leq t$. This, along with the fact that at the start $A[1 \cdots i]$ was sorted, leads us to $A[1 \cdots j - 1] \leq t$.

As we have not taken proper care of the end condition, the guard in the above loop will cause a subscript error when $j = 1$. So we modify it to

$$j \neq 1 \textbf{ cand } A[j - 1] > t$$

where **cand** is similar to **and** but the second operand is evaluated only if $j \neq 1$ (*C* has a similar operator). This allows for a simple high level design.

- shift $A[j-1]$ to the right is refined as

$$A[j] := A[j-1]$$

Collecting all the refinements we get

```
i := 1
do i ≠ n → t := A[i + 1]; j := i + 1
           do j ≠ 1 cand A[j − 1] > t → A[j] := A[j − 1]
                                        j := j − 1
           od
           A[j] := t
           i := i + 1
od
```

We conclude this section with some comments on program maintenance and efficiency. Program maintenance, i.e., program modification that is due to changing specifications or in response to design errors, should be carried out by making changes in the refinements and not just the final program. Making changes in only the final program renders the design (i.e., refinements) obsolete; consequently, an updated version of the design will no longer exist and subsequent program maintenance becomes increasingly difficult.

When program specifications change, start from the initial refinement and locate the refinement affected. Modify this refinement and carry the effects of this change down to the last level of refinement.

When a design error is detected, locate the most abstract refinement in which the design error was first made. Then carry the change caused by the removal of the design error down to the final refinement.

A program that does not have the desired efficiency (i.e., performance) characteristics must be redesigned. We locate the most abstract refinement R where a design decision was made that resulted in these characteristics. A proper design modification from refinement R onwards leads to the desired efficiency characteristics. Predecessors of refinement R remain unchanged.

IV. RECURSION

Stepwise refinement and recursion blend naturally with each other. Many programmers avoid recursion and treat it as a novelty.[10] Some problems are best expressed recursively, even though languages like FORTRAN and COBOL are not recursive, and this inhibits programmers from thinking and designing recursively. Also, the examples of recursion in text books, e.g., factorial, Fibonacci numbers, etc., are not convincing about its utility.

Efficiency has often been cited as a reason against using recursion. In these days of increasing software costs and decreasing hardware costs, this reason is not very convincing. It is better to have a recursive design that is simpler, easier to understand, and easier to show correct than a corresponding nonrecursive version. If efficiency is still a criterion, then the recursive design can be systematically transformed into a nonrecursive one.[11] The following examples illustrate the development of recursive programs:

1. Write a procedure to print a binary tree with root R (Fig. 4). Each node is of the form

VALUE	
LEFT	RIGHT

where

(*i*) VALUE is the data at the node,
(*ii*) LEFT, RIGHT are the pointers to the subtrees,
(*iii*) a NIL pointer value denotes the absence of a subtree.

Initial refinement P_0: Print binary tree with root R

P_1:

if $R \neq$ NIL → Print binary tree with root LEFT(R)
{left subtree}
Print VALUE(R)
Print binary tree with root RIGHT(R)
{right subtree}
[]R = NIL → skip
fi

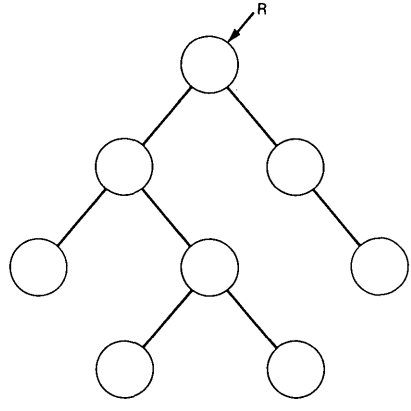


Fig. 4—A binary tree with root R.

The notation $p(x)$ denotes the component p of the node pointed to by x. Writing the above in Pascal we get:

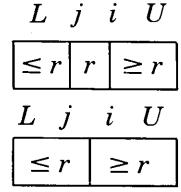
```
procedure print (R: ↑ node);
  begin
    if R ≠ nil
      then begin print(R ↑. LEFT);
                 writeln(R ↑. VALUE);
                 print(R ↑. RIGHT)
                 end
    end
```

2. This example illustrates a fast sorting technique called quicksort (Hoare[12]). Array A, with bounds L and U ($L \leq U$), is to be sorted in nondecreasing order.

Initial refinement P_0: Quicksort(A, L, U).

P_1:

if one element → skip
[ ]two elements → order them
[ ]more than two elements → Partition A such that



(at least one element per partition in this case)
where r is an arbitrary value
Quicksort(A, L, j)
Quicksort(A, i, U)

fi

P_2:
- one element
  is refined as
  $U - L = 0$
- two elements is refined as $U - L = 1$
- order them
  is refined as

  if $A[U] \leq A[L]$ → swap($A[L]$, $A[U]$)

  [] $A[U] \geq A[L]$ → skip

  fi
- more than two elements
  is refined as
  $U - L > 1$

• Partition A such that $\cdots$
is refined as

$r := A[(U + L) \div 2]$
$i := L; j := U$ {$A[L \,..\, i-1] \leq r$ and $A[j+1 \,..\, U] \geq r$ – Invariant I}
do $i < j \;\rightarrow$ Extend left partition by increasing i
Extend right partition by decreasing j
Rearrange elements so that invariant I is restored
od

P_3:
• Extend left partition $\cdots$
do $A[i] < r \;\rightarrow\; i := i + 1$ **od** {$A[i] \geq r$}
• Extend right partition $\cdots$
do $A[j] > r \;\rightarrow\; j := j - 1$ **od** {$A[j] \leq r$}
• Rearrange $\cdots$
if $i < j \;\rightarrow$ swap ($A[i]$, $A[j]$)
$i := i + 1; j := j - 1$
[] $i \geq j \;\rightarrow$ skip
fi

V. MULTIVERSION PROGRAMS

A set of programs is said to constitute a program family if it is worth while to study programs from the set by first studying the common properties of the set and then determining the special properties of the individual family members. A typical family is the set of versions of an operating system distributed by a manufacturer.[13] Such a family is also called a set of multiversion programs. Stepwise refinement enables multiversion programs to be developed conveniently and naturally.

Multiversion programs may be built for the following reasons:[13,14]

(*i*) Economics. It is cheaper to build one program and then modify it to get another version than it is to build the second program from scratch.

(*ii*) Experimentation. Experimental prototypes may be built to study the feasibility of building a particular system. The experimental versions along with the final program constitute the multiversions.

(*iii*) Faulty program design. Another version of the program is built to correct the design faults of a prior version.

Classically, multiversion programs have been built by first building one working version of a program. Another version is built by modifying this program and so on, as shown in Fig. 5. A set of multiversion programs produced as in Fig. 5 has one common ancestor. According to Parnas,[13] it is common for the descendants of one program to share some of their ancestors' characteristics which are not appropriate to the descendants. In building the earlier version, some decisions were

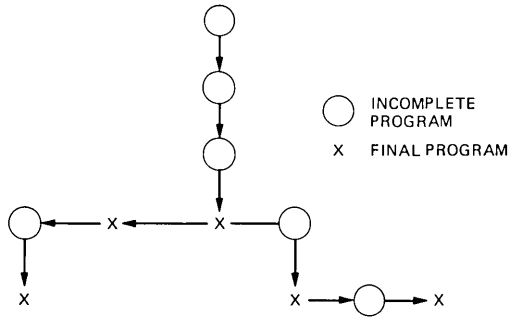


Fig. 5—Traditional way of building multiversion programs.

made which would not have been made in the descendant programs had they been built independently. Removal of these decisions entails a lot of reprogramming. Consequently, programs have performance deficiencies because they contain decisions not really suitable for them. To build another program version, the program must first be complete and working. Relevant changes in an ancestor program that are not reflected in the descendant program cause maintenance problems.

Stepwise refinement allows us to develop multiversion programs without the above problems. Never modify a complete program; always begin from one of the intermediate refinements which does not contain any design decisions unsuitable for the new version. This process is illustrated in Fig. 6.

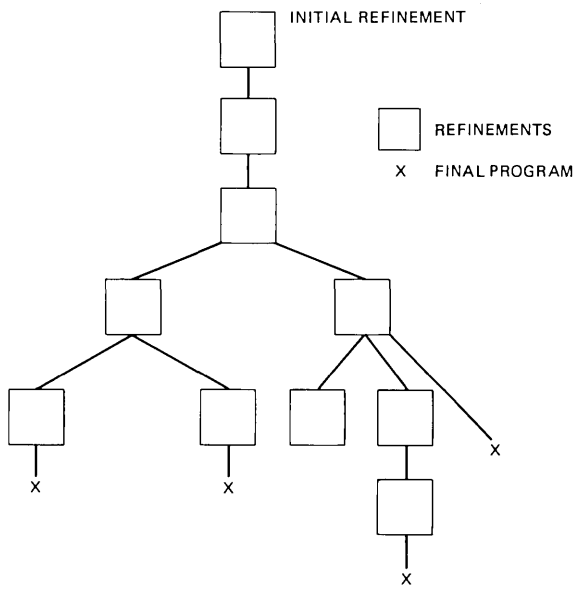


Fig. 6—Multiversion program development by stepwise refinement.

All decisions made above a branch point are shared by the descendants. Refinements are developed so that common decisions of multiversion programs are above the branch point. A branch point results when two or more versions require different strategies (e.g., storage management techniques).

Also refinements below a branch point may be carried out in parallel—we do not have to wait until a working program is available.

VI. SUGGESTIONS FOR REFINEMENT

1. Develop the program in a gradual sequence of steps.
2. In each step, refine one or more instructions of the given refinement.
3. Terminate the refinement process when the instructions have been expressed in the desired programming language or when they can be mechanically translated to the programming language.
4. Use information about the problem and its domain in the formulation of abstract instructions.
5. Use notation natural to the problem domain.
6. Make up abstract instructions as desired. However, they must eventually be translatable to an executable form.
7. Make refinements reflect the instructions they represent in detailed form.
8. Be aware that every refinement represents some implicit design decision and consider alternate solutions. Keep a written record of the major decisions made along with the refinements.
9. Use recursion when appropriate. Even if the language does not support recursion, recursive solutions should still be considered. If recursive solutions are selected, they can be systematically converted to nonrecursive solutions.
10. Use data refinement along with instruction refinement.
11. Postpone representation of data as long as possible. This minimizes modifications to the design when an alternate representation is to be used.
12. If an instruction appears more than once, use a procedure call; refine the instruction only once. Use procedure calls when they clarify program structure.
13. Recognize abstract data types and separate their refinements from the rest of the program, i.e., do not refine the data type operations in line—use procedure calls.
14. Try to use loop invariants to develop loops; they give a better idea of the instructions in the loop body and the guards.
15. If a refinement solution does not turn out to be appropriate, repeat the refinement process using the additional knowledge derived from the previous attempt; stepwise refinement is an iterative process.

VII. TOPICS RELATED TO STEPWISE REFINEMENT

In this section, the idea of abstract data types is explained along with the concept of data refinement. This is followed by a discussion of the formal specifications of abstract data types. We then show how stepwise refinement may help simplify program correctness proofs. Finally, we briefly argue that stepwise refinement can be used in developing parallel programs.

7.1 Abstract data types

A data type is not only a set of values but also the operations that can be performed on them.[15] A primitive data type is a data type that is available in the programming language. An abstract data type is a data type not available in the programming language and is implemented in terms of other abstract and primitive data types.

The implementation of a data type consists of three parts—storage allocation, initialization, and the definition of operations. Consider the following implementation of the data type integer stack of size 100 in PL/I:

(*i*) storage allocation

```
DCL (S(100)
    ,NS /*NUMBER OF ELEMENTS IN S*/
    )FIXED BIN;
```

(*ii*) initialization

```
NS = 0;
```

(*iii*) operations

```
PUSH:PROCEDURE(A, NA, X);
     DCL (A(100), NA, X) FIXED BIN;
     IF NA = 100
        THEN PUT SKIP LIST ('OVERFLOW ERROR');
        ELSE DO; NA = NA + 1; A(NA) = X; END;
     END PUSH;

POP:PROCEDURE(A, NA);
    DCL (A(100), NA)FIXED BIN;
    IF NA = 0
       THEN PUT SKIP LIST ('UNDERFLOW ERROR');
       ELSE NA = NA - 1;
    END POP;

TOP:PROCEDURE(A, NA) RETURNS (FIXED BIN);
    DCL (A(100), NA)FIXED BIN;
    IF NA = 0
       THEN PUT SKIP LIST('ERROR-STACK EMPTY');
       ELSE RETURN(A(NA));
    END TOP;
```

```
EMPTY: PROCEDURE(A, NA)RETURNS(BIT(1));
       DCL (A(100), NA)FIXED BIN;
       RETURN(NA = 0);
       END EMPTY;
```

Such implementations of abstract data types suffer from many disadvantages. Representation details are not hidden from the programmer. This leads to the following:

(*i*) Representation dependent programming, perhaps for "efficiency" reasons. For example, the programmer may directly inspect the top of the stack instead of using the procedure TOP; a change in the representation will then require changes in the program.

(*ii*) Violation of the specifications of the abstract data type. For example, in case of the stack the programmer may delete an element in the middle of the stack. If such an ability is desired, then the specifications should be changed.

(*iii*) Inadvertent or malicious violation of the integrity of the abstract data type. For example, *NS* could be set to 0 even if the stack is not empty.

In addition, the programmer has to be concerned about which components of the representation have to be passed as arguments to the procedures associated wth the abstract data type. To be uniform, we have passed all the components of the stack representation for every operation.

Modern programming languages such as CLU[16-18] and Alphard[1] provide data abstraction features, called clusters and forms, respectively, *that remove the above problems.* In addition, they support *data refinement.* We use CLU's clusters to illustrate data refinement and give an example of programming with abstract data types. The notation used is similar to that of CLU.

```
stack = cluster is push, pop, top, empty;
  rep = record [ns:integer;
        s: array[1 .. 100] of integer]
  create = oper( ) returns cvt;
      s:rep
      s.ns:=0
      return s
      end
  push = oper(a:cvt, x:integer)
    if a.ns = 100
    then overflow error
    else begin a.ns := a.ns + 1
              a.s[a.ns] := x
              end
```

```
        end
    pop = oper(a:cvt)
      if a.ns = 0
            then underflow error
                 else a.ns := a.ns − 1
        end
    top = oper(a:cvt) returns integer
        if a.ns = 0
              then stack empty error
              else return a.s.[a.ns]
        end
    empty = oper(a:cvt) returns boolean
          return a.ns = 0
          end
end stack
```

Having defined the stack cluster, the programmer can declare variables of type stack, e.g., *b:stack*.

The first line of the stack definition states that the operations available to stack users are push, pop, top, and empty. The operation must be prefixed by the type name, e.g., the special operation **create** is automatically executed when a variable of type stack is declared.

The line beginning "**rep** = " specifies that a stack is represented by an integer array and an integer. This information cannot be used outside the cluster, thus making the rest of the program representation independent.

The special symbol **cvt** (convert) means that the variable is of the abstract type outside the operation and of the representing type inside the operation.

An operating system example

Suppose we are writing an operating system. Jobs are to be scheduled according to their priority (10 being the highest and 1 the lowest). The next job to be executed is the one with the highest priority. If there is more than one job with the highest priority, then the one selected for execution is the one that waited the most (FIFO):

```
begin {operating system}
    .
    .
    .
Add job j with priority p to the list of
      jobs waiting for execution
    .
    .
    .
Wait until there is a job to execute
let j be the next job to be executed
    .
    .
    .
end {operating system}
```

To implement the job scheduling we define a cluster called spq (system of priority queues), with operations add, empty , and next job.

```
spq = cluster is add, empty, nextjob
  rep = array[1 .. 10] of queue
  create = oper(    ) returns cvt
      s:rep
      return s
      end
add = oper(s:cvt, job:integer, prty:1 .. 10)
    queue$add(s[prty], job≠)
    end
empty = oper(s:cvt) returns boolean
    i: integer := 0
    while i ≠ 10 do
        begin i := i + 1
        if queue$empty(s[i]) then return false
        end
    return true
    end

nextjob = oper(s:cvt) returns integer
    i:integer := 11
    j:integer
    while i ≠ 1 do
        begin i := i − 1
            if ~ queue$empty(s[i])
              then begin j := queue$front(s[i])
                        queue$delete(s[i])
                        return j
                        end
        end
    end
end spq
```

The cluster spq is implemented in terms of the abstract data type queue. For example, cluster spq's operation add uses cluster queue's operation add, i.e., queue$add. We refine instructions of the abstract data type queue by implementing a queue cluster. For this example, we assume that no more than 50 jobs of the same priority will be waiting at the same time. We shall use a wrap-around array representation for the queue in which

(*i*) an array of size 51 is used; only 50 elements can be stored in the queue,

(*ii*) $F = L$ means that the queue is empty,

(*iii*) **mod**(F, 51) + L points to the next element in the queue,
(*iv*) **mod**(L, 51) points to the last element in the queue,
(*v*) **mod**(L, 51) + 1 = F means that the queue is filled.

```
queue = cluster is delete, add, front, empty;
        rep = record[a:array[1 .. 51] of integer; F, L:integer]
        create = oper(  )returns cvt
            q:rep
            q.F := 0
            q.L := 0
            end

    delete = oper(q:cvt);
        if q.F = q.L
                then error-empty queue
                else q.F := mod(q.F, 51) + L
            end
    add = oper(q:cvt, j:integer)
        if mod(q.L, 51) + 1 = q.F
            then error-queue full
            else begin q.L = mod(q.L, 51) + 1
                       q.a[q.L] := j
                       end
        end

front = oper(q:cvt) returns integer
    if q.F = q.L
        then error-empty queue
        else return q.a[mod(F, 51) + 1]
    end

empty = oper(q:cvt) returns boolean
    return q.F = q.L
    end
end queue
```

7.2 Formal specifications of abstract data types

In this section, we consider the formal specifications of abstract data types. In particular, we take a brief look at the algebraic specification technique proposed by Guttag.[19, 20] Abstract data types are implemented in terms of other data types by the refinement or decomposition of specifications.

The algebraic specifications consists of two parts:

(*i*) the syntax—here the operations of the data type are listed indicating the number of arguments, the argument types, and the result type.

(*ii*) the semantics—here axioms are given that relate the values created by the operations.

In the basic notation which we will use, the operations are functions without side effects; none of the arguments are changed. Guttag[21] has extended the notation to allow for changes in arguments, i.e., to allow procedures.

We shall present algebraic specifications for the abstract data type stack considered earlier. We shall then give specifications for an array. Finally we shall refine the stack operations in terms of array operations. To keep the specifications as simple as possible, we shall consider unbounded (i.e., infinite size) stacks and arrays.

Stack specifications:

```
1.    type stack
2.        syntax
3.            create( ) → stack
4.            push(stack, integer) → stack
5.            pop(stack) → stack
6.            top(stack) → integer
7.            empty(stack) → boolean
8.        semantics
9.            declare s:stack; x:integer
10.           pop(create( )) = underflow error
11.           pop(push(s, x)) = s
12.           top(create( )) = empty stack error
13.           top(push(s, x)) = x
14.           empty(create( )) = true
15.           empty(push(s, x)) = false
```

Line 3: specifies the syntax of the create operation. The result of calling create, which has no parameters, is an object of type stack.

Line 4: operation push takes as input a parameter of type stack and a parameter of type integer. It returns a value of type stack.

Line 10: The result of applying the pop operation on a stack that has just been created is an underflow error. "=" is the equality operator (not assignment).

Line 11: The result of popping a stack s on which the last operation was to push a value x is the initial stack s.

These specifications specify the abstract data type completely. For details on how to construct the specifications, see Ref. 20. We now give the specification for the type array which will be used to implement the type stack.

```
1.    type array
2.        syntax
```

```
3.          createarray( ) → array
4.          assign(array, integer, integer) → array
5.          access(array, integer) → integer
6.      semantics
7.          declare a:array; i, j, x:integer
8.          access(createarray( ), i) = undefined error
9.          access(assign(a, i, x), j)
                    = if i = j
                        then x
                        else access(a, j)
```

Operation assign(a, i, x) stands for an array whose ith element has been assigned the value x. Operation access(a, i) stands for the ith element of a.

We now implement the abstract data type stack by decomposing stack operations in terms of array operations. We give axioms called programs that give the effect of stack operations in terms of array operations.

```
1.    stack implementation
2.        syntax
3.            STK(array, integer) → stack
4.        programs
5.            declare a:array, t, x:integer
6.            create( ) = STK(createarray( ), 0)
7.            push(STK(a, t), x) = STK(assign(a, t + 1, x),
                                                t + 1)
8.            pop(STK(a, t)) = if t = 0
                                then underflow error
                                else STK(a, t − 1)
9.            top(STK(a, t)) = if t = 0
                                  then empty stack error
                                  else access(a, t)
10.           empty(STK(a, t)) = (t = 0)
```

Given the formal specifications for an abstract data type, an initial inefficient implementation, called the direct implementation, can be automatically generated.[19] Thus one can test some facets of a high-level data type before fixing upon a particular implementation. Thus a true top-down implementation methodology can be achieved.

7.3 Program correctness

Stepwise refinement provides a natural environment for reducing the problem of showing the correctness of a large program into showing the correctness of several smaller programs.

A program is said to be correct if it meets its input and output

specifications (which may include performance criteria). Alternately, a correct program is one that transforms a state (i.e., data values) representing the input specifications into one representing the output specifications. A program is thus viewed as a specification transformer (Dijkstra[6] calls it a predicate transformer). Let

(*i*) I and O be the input and output specifications for the problem being solved.

(*ii*) P_0 be the initial problem formulation and P_0^* the corresponding final program.

Then we say that P_0 has been solved correctly if P_0^* is correct with respect to I and O. If P_0 is a nontrivial problem, then proving the correctness of P_0^* will be correspondingly nontrivial.

Stepwise refinement allows the correctness proof of a program to be reduced to the correctness proofs of smaller programs. Suppose P_0 is refined or decomposed into the subproblems $P_{10}, P_{11}, \cdots, P_{1n}$, with each P_{1i} having specifications S_i and S_{i+1} ($S_0 = I$ and $S_{n+1} = 0$). The problem of proving P_0^* correct is now reduced to proving P_{1i}^* ($0 \leq i \leq n$) correct, where P_{1j}^* is the program corresponding to P_{1j}. Owicki[22] provides an example of such a correctness proof.

In summary, stepwise refinement provides a natural medium for a difficult proof to be decomposed into several smaller proofs. A proof is any convincing demonstration of a program's correctness. However, the conventional approach to understanding programs in terms of how computers execute them is inadequate. A more mathematical approach is needed even if it is used informally.[23] Alagic[24] contains many examples of programs designed with correctness proofs in mind.

7.4 Parallel programs

The development of parallel programs is no different than the development of sequential programs as far as stepwise refinement is concerned. Instead of using only sequential constructs, like **begin** S_1, S_2; $\cdots$; s_n **end** in Pascal, we now use constructs for parallel programming,[25, 26] as shown below:

(*i*) **cobegin** $S_1, S_2, \ldots, S_n$ **coend**

The statements $S_1, S_2, \ldots, S_n$ are executed in parallel.

(*ii*)
when $b_1 \rightarrow SL_1$
[]$b_2 \rightarrow SL_2$
⋮
[]$b_n \rightarrow SL_n$
end

Wait till one of the guards b_i is true and then execute the corresponding statement list

(*iii*) **cycle** $b_1 \rightarrow SL_1$
$[]b_2 \rightarrow SL_2$
⋮
$[]b_n \rightarrow SL_n$
end

Endless repetition of a when statement.

If several guards are true within a **when** or a **cycle** statement, then one of the corresponding statement lists is executed nondeterministically.

VIII. CONCLUSIONS

In this tutorial, we have tried to illustrate the stepwise refinement technique, its advantages, and related topics. Stepwise refinement can be learned easily with some practice. It blends in naturally with the newer concepts in programming languages and methodology (e.g., abstract data types, parallel programming, etc.).

Stepwise refinement does not provide a solution to the problem. No methodology, old or new, is going to discover algorithms (i.e., problem solutions) for the programmer. The algorithms must come from the programmer's education, experience, and ingenuity.

Stepwise refinement encourages the development of a problem solution in a systematic fashion that is easy to understand, modify, and improve upon. The various refinements should not be discarded once the final program version is arrived at. They are part of the program documentation. Understanding the final program without them is hard even if the program is small (e.g., the eight-line final program version of insertion sort in Section III).

The reader is urged to try stepwise refinement on some problems, especially large ones.

IX. ACKNOWLEDGMENTS

I am very grateful to the following people for their constructive and critical comments, which were very valuable. They are (in alphabetical order) A. P. Boysen, Jr., R. H. Canaday, D. G. Dzamba, A. R. Feuer, T. B. Muenzer, and D. A. Nowitz.

REFERENCES

1. W. A. Wulf, "Alphard: Toward a Language to Support Structured Programs," Computer Science Dept. report, Carnegie-Mellon University, Pittsburgh, Penn., April 1974.
2. F. P. Brooks, *The Mythical Man-Month*, Reading, Mass.: Addison-Wesley, 1975.
3. C. Alexander, *Notes on the Synthesis of Form*, Boston; Harvard University Press, 1970.

4. N. Wirth, "Program Development by Step Refinement," Commun. ACM, *14*, No. 4 (1971).
5. K. Jensen and N. Wirth, *Pascal User Manual and Report*, New York: Springer, 1974.
6. E. W. Dijkstra, *A Discipline of Programming*, New Jersey: Prentice Hall, 1977.
7. M. A. Jackson, "Information Systems: Modelling, Sequencing, and Transformations," Proc. Third Int. Conf. Software Engineering, Atlanta, Ga., 1978.
8. D. Gries, "A Note on Iteration," TR77-323, Department of Computer Science, Cornell University, Ithaca, NY.
9. G. D. Bergland, private communication.
10. D. Gries, "Recursion as a Programming Tool," TR75-234, Department of Computer Science, Cornell University, Ithaca, NY.
11. S. Sickel, "Removing Redundant Recursion," Technical Report, Information Sciences, University of California, Santa Cruz, Calif., 1978.
12. C. A. R. Hoare, "Quicksort," Comput. J., *5*, No. 1 (1962).
13. D. Parnas, "On the Design and Development of Program Families," IEEE Trans. Software Eng., March 1976.
14. R. H. Canaday, private communication.
15. J. H. Morris, Jr., "Types are Not Sets," ACM Symp. Princ. Prog. Lang., Boston, Mass., 1973.
16. B. Liskov and S. Zilles, "Programming with Abstract Data Types," Proc. SIGPLAN Symp. Very High Level Lang., Santa Monica, Calif., 1974.
17. B. Liskov, "A Note on CLU," Computation Structures Group Memo 112, MIT Project MAC, Cambridge, Mass., November 1974.
18. B. Liskov et al., *CLU Reference Manual*, Cambridge, Mass.: MIT Laboratory for Computer Science, 1978.
19. J. V. Guttag et al., "Abstract Data Types and Software Validation," Commun. ACM, *21*, No. 12 (1978).
20. J. V. Guttag, "The Algebraic Specification of Abstract Data Types," Acta Inform., *10* (1978).
21. J. V. Guttag et al., "Some Extensions to Algebraic Specifications," Proc. Lang. Design for Reliable Software, March 1977.
22. S. Owicki, "The Specification and Verification of a Network Mail System," CSL TR-159, Computer Science Lab. Report, Stanford, California (1979).
23. D. Gries, "On Believing Programs to be Correct," Commun. ACM, *20*, No. 1 (1977), pp. 49, 50.
24. S. Alagic and M. A. Arbib, *The Design of Well-Structured and Correct Programs*, New York: Springer, 1978.
25. P. Brinch Hansen, "Structured Multiprogramming," Commun. ACM, *15*, No 7 (1972).
26. P. Brinch Hansen, "Distributed Processes: A Concurrent Programming Concept," Commun. ACM, *21*, No. 11 (1978).

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 3, March 1981
Printed in U.S.A.

A New Code for Transmission of Ordered Dithered Pictures

By O. JOHNSEN

(Manuscript received September 22, 1980)

This paper presents a new predictor for coding dithered pictures. A dithered picture, a two-tone picture which gives the illusion of a picture with many shades of grey, is obtained by comparing the grey-level picture with a position-dependent threshold. When the intensity of a picture element (pel) exceeds the threshold, it is classified as white; otherwise it is black. In the new prediction scheme, the color of a pel is predicted from pels having nearly the same threshold level. Therefore, the position of the pels used for prediction varies according to the threshold level. Computer simulations show that prediction errors are reduced by 50 percent for certain classes of originals, and the entropy is reduced by 20 percent compared to the result obtained with a previous predictor. An advantage of this new prediction scheme is that it appears to be less sensitive to picture content.

I. INTRODUCTION

Dithering is an image processing technique that creates a two-level picture that gives the illusion of a multilevel picture by appropriately arranging the spatial density of the two levels (usually black and white) on the picture.[1-5] The dithering technique consists of comparing a multilevel image with a position-dependent threshold and setting pels to white when the input signal exceeds the threshold. Other pels are set to black. The matrix of threshold values (called the dither matrix) is repeated over the entire picture to provide the threshold pattern for the whole image.

The merit of a dither matrix is judged from the quality of its rendition of the original picture. A class of dither matrices of special interest is the ordered dither matrices,[3] which use a simple recursion algorithm to create dither matrices of size $2^n \times 2^n$ to simulate $2^{2n} + 1$ brightness levels.

In the case of the 4 × 4 matrix, the 16 threshold levels are put in the following positions:

$$\begin{vmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 5 \end{vmatrix}. \tag{1}$$

The ordered dither matrices have the advantage of rendering grey level pictures with as good a subjective quality as with empirical dither matrices.[3] Moreover, we show that ordered dither matrices are especially convenient for the proposed new predictor. Figure 1 shows three dithered pictures, "Karen," "Engineer drawing," and "House," that are used for computer simulations. The originals for these pictures are 10 cm by 10 cm and were scanned to generate an array of 512 by 512 pels. The pels were digitized with a uniform 8-bit PCM code (256 levels). Matrix 1 then becomes

$$\begin{vmatrix} 8 & 136 & 40 & 168 \\ 200 & 72 & 232 & 104 \\ 56 & 184 & 24 & 152 \\ 248 & 120 & 216 & 88 \end{vmatrix}. \tag{2}$$

Note that Figs. 1a, b, and c were made from digitizing the same originals as used in Refs. 4 and 6 but not from the same digitized versions.

Efficiently coding the bits of the dithered image reduces the data rate. If the pels of the dithered picture are sampled in a typical raster scan fashion, the frequent alternations between black and white pels prevent the direct use of run-length coding, which is an efficient redundancy reduction scheme.[4] Others have proposed a modification to the straightforward raster scan sampling or to the direct use of the picture pels. Judice proposed a different sampling order, a bit interleaving code.[4] Netravali et al.[6] devised a predictive coding technique that is an extension of their two-level facsimile coding scheme.[7] In both cases, the binary picture containing the interleaved bits or the prediction errors can be efficiently run-length coded. In Netravali's coding technique, the prediction of the value of a pel (1 ≙ white, 0 ≙ black) is made dependent on the values of four neighbor pels as well as to the threshold level. Figure 2 shows the ordered dither matrix of (2) and the prediction that he used. With a dither matrix of 16 threshold levels, and a prediction from 4 neighboring pels, the prediction table has 256 possible states. This prediction scheme takes advantage of the correlation between adjacent pels, but it can only partially exploit the much stronger correlation that exists between neighboring pels with the same or similar threshold level.

The scheme proposed in this paper predicts a pel according to the

Fig. 1a—Dithered test picture of Karen.

value of nearby pels of the same or similar threshold level. The new predictor gives for two of the three test pictures 50 percent fewer prediction errors compared to the previous one, and 10 percent fewer prediction errors for the third test picture.

II. PREDICTION PRINCIPLES AND ALGORITHMS

2.1 Prediction principle

A pel can best be predicted from the following information:

(*a*) the threshold level of the pel itself,

(*b*) the values of the nearest (already coded) pels with similar threshold levels.

We use four previous pels for prediction and consider pictures which have been dithered using the 4 × 4 ordered dither matrix of (2). There are then 16 threshold levels and since each pel used for prediction has only two possible values, 256 states are defined by all the possible

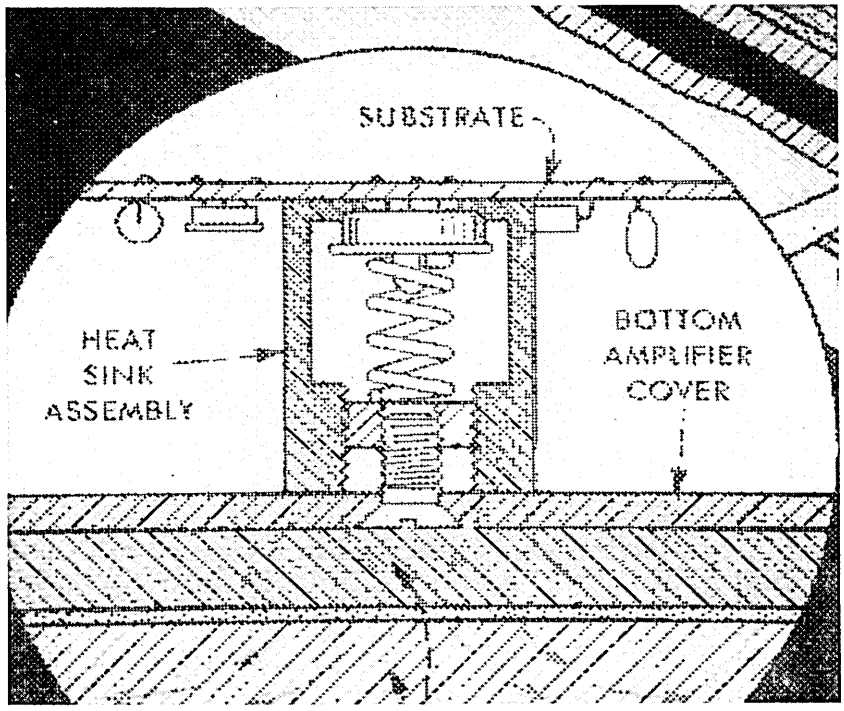


Fig. 1b—Dithered test picture of Engineering drawing.

combinations of the predictor and of the threshold level. The predictor code book contains 256 bits, each indicating whether the pel is predicted to be black or white. The code book is simply obtained by statistical measurement. From these predictions, we generate a new picture where the pels correctly predicted are put to "0" while the pels incorrectly predicted are put to "1." The new picture, called the "error picture," can easily be run-length coded and transmitted efficiently. Knowledge of the predictor code book allows the receiver to reconstruct the original dithered picture. The four pels used for prediction are:

(a) the two nearest pels with the same threshold level,
(b) the nearest pel with the next higher threshold level,
(c) the nearest pel with the next lower threshold level.

For cases (b) and (c), when two pels satisfy the rule at the same time, the pel with the smallest correlation with the other prediction pels is chosen. Also, for case (b), if there are no pels with a higher threshold level, a second pel with the next lower threshold level is used for prediction. A similar situation can appear in case (c).

2.2 Prediction algorithm

Let the pel to be predicted, S_{ij}, have position $(i; j)$, where i is the line number, j is the position on the line, and i and j are increasing down and to the right, respectively. The two nearest pels with the same threshold level naturally have coordinates $(i - 4; j)$ and $(i; j - 4)$ since a 4×4 dither matrix is used.

To find the two other pels used for prediction, we must look at the structure of the 4×4 ordered dither matrix. The procedure is described in the appendix. We give here only the result, showing that the coordinates of the four pels used for the prediction are

1. $i; j - 4$,
2. $i - 4; j$,
3. $i - 2; j + 2$,
4. variable.

The position of the prediction pel with variable position is given according to the position of *Sij* within the dither matrix by

Fig. 1c—Dithered test picture of House.

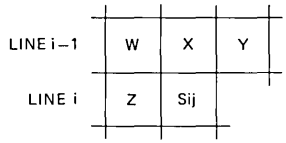


Fig. 2—Picture elements used for prediction, where *Sij* is the picture element to be predicted, and *W*, *X*, *Y*, and *Z* are picture elements used to predict *Sij*.

$$\begin{vmatrix} i-2;j-2 & i-1;j & i-2;j & i-2;j \\ i-3;j+1 & i-3;j-1 & i-2;j & i-2;j \\ i-1;j+1 & i-1;j-1 & i-2;j & i-2;j \\ i-2;j-2 & i-3;j & i-2;j & i-2;j \end{vmatrix} \quad (3)$$

Figure 3 shows the positions of the four pels used for the prediction.

Note that for the eight pels in the two right rows of the matrix the prediction pel with variable position is the same.

This new predictor can be called a position-dependent predictor since one of the pels used for the prediction is variable. In the case where using a position-dependent predictor is undesirable, the predictor size can be reduced and only three pels can be used, or the fourth pel used in the prediction can be fixed to the most-used position ($i - 2; j$). Both cases lead to a slight increase in the number of prediction errors.

2.3 Extension to other dither matrices

The same prediction principle can be applied to different dither-matrices or different sizes of matrices, but with care. For example, when applying the same rule to an ordered 8 × 8 dither matrix, three of the four pels of the predictor are fixed and their locations, compared to the pel to be predicted, are in the same direction but twice as far as

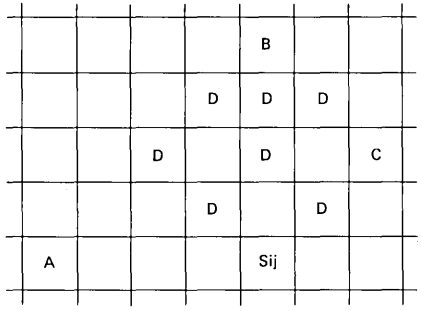


Fig. 3—Position of the pels used for the prediction, where *Sij* is the pel to be predicted, *A*, *B*, and *C* are the three fixed pels used for predicting, and the *D*'s are the different positions occupied by the pel with variable position used for prediction.

for a 4 × 4 matrix. Their locations are

1. $i; j - 8$,
2. $i - 8; j$,
3. $i - 4; j + 4$.

The prediction pel with variable position again has the same relative position for the right half of the matrix, the coordinates being $(i - 4; j)$. For the other half, the position is variable, but often the same within each 2 × 2 submatrix. The position matrix of the prediction pel with variable position can be constructed by looking at the structure of the 2 × 2 and 4 × 4 submatrices.

It is questionable whether such a predictor gives good prediction. The prediction is now made according to the value of the pels twice as far away, as in the case of a 4 × 4 dither matrix. The error rate is therefore likely to increase. The best solution is to use the same prediction algorithm as for the 4 × 4 dither matrix, since the 8 × 8 dither matrix is composed of four 4 × 4 dither matrices whose threshold levels are only very slightly different. The number of prediction errors would then be only slightly higher than if the picture was dithered with a 4 × 4 matrix. A slight decrease in prediction errors can be obtained by considering all 64 threshold levels instead of 16 for constructing the prediction table. The prediction table would then contain 1024 states instead of 256.

In the case of other dither matrices (nonordered) the prediction principle is the same but the algorithm is different, since the position of the nearest pels with similar threshold level changes. In most cases, two of the four pels used for prediction would have a variable position, thus showing the advantage of using the ordered dither matrices with this prediction scheme.

This prediction scheme can be extended by adding more pels to the predictor; for example, using pels closer to the pel to be predicted, thereby decreasing the number of prediction errors along a sharp edge.

III. SIMULATION RESULTS

Computer simulations are performed using the three dithered pictures of Fig. 1. The results are compared to those obtained using the technique proposed by Netravali et al.[6] Two measures of performances are made: counting the number of prediction errors and measuring the entropy of the run-length statistics of the picture containing the prediction errors. The entropy is converted into bits per pel. The run-length entropy measured is the classical run-length entropy, given for example in Ref. 6.

Table I gives the number of prediction errors with the new predictor. For comparison, the number of prediction errors with the technique in Ref. 6 is also given. Compared with the predictor of Ref. 6, the number

of prediction errors is reduced by 53 percent for "Karen" and "House" and 10 percent for "Engineering drawing."

Table II gives the entropy comparisons. Four entropies are given. The first is the entropy of the run-length statistics of the prediction error when using separate prediction tables for each picture. The second is the same entropy in the case where a single prediction table is used for all three pictures (the prediction table is optimized to the sum of the three pictures).

The third entropy is the entropy of the run-length statistics of the prediction errors when the prediction errors are ordered according to the probability of error[6] (good-bad ordering). In the good-bad ordering the prediction errors of the pels of a line are filled in the right side of a line if their error probabilities are high, while they are put on the left side when their error probabilities are low.[6] The goodness threshold used is 0.05.

The fourth entropy is the same as the latter but with a single prediction table for all three pictures.

The entropy measurements show the great improvements obtained by this new prediction technique compared to the results from Ref. 6. For "Karen" and "House" the reduction in the run-length entropy is 25 to 30 percent, but it is limited to 3 percent for "Engineering drawing." When a single prediction table is used, the reduction in entropy is 28 percent for "Karen" and "House" but it reaches 10 percent for "Engineering drawing." In the case of good-bad ordering, the decrease in entropy with the new prediction technique compared to Ref. 6 is 17 to 24 percent for "Karen" and "House" while an increase of 2 percent appears for "Engineering drawing." When a single prediction table is used for all pictures, the decrease is 24 percent and 17 percent for "Karen" and "House" and 3 percent for "Engineering drawing."

The gain obtained with the new predictor for "Karen" and "House" is very different for "Engineering drawing." It can be explained by the picture characteristics. The "Karen" and "House" pictures are really half-tone pictures with mostly gradual changes in brightness while "Engineering drawing" is a graphical picture containing mostly steep

Table I—Comparison of the number of prediction errors

Pictures	Prediction Errors with New Predictor	Prediction Error with Predictor of Netravali
Karen	12,613	26,557
Engineering drawing	25,742	28,526
House	12,106	26,156

Table II—Comparison of entropies (in bits/pel)

	Karen	Engineering Drawing	House
I. New Predictor			
Run-length coding with separate prediction table for each picture	0.221	0.396	0.214
Run-length coding with a single prediction table	0.228	0.403	0.216
Run-length coding with good-bad ordering with separate prediction table for each picture	0.196	0.376	0.187
Run-length coding with good-bad ordering and a single prediction table	0.202	0.380	0.191
II. Predictor of Netravali			
Run-length coding with separate prediction table for each picture	0.317	0.409	0.286
Run-length coding with a single prediction table	0.317	0.446	0.303
Run-length coding with good-bad ordering and separate prediction table for each picture	0.259	0.367	0.225
Run-length coding with good-bad ordering and a single prediction table	0.278	0.393	0.253

changes in grey level. These steep changes are well predicted with Netravali's predictor and therefore practically no improvements are obtained with the new predictor.

An advantage of this new technique is that the predictor is quite independent of the characteristics of individual pictures. The entropy with the predictor optimized for the sum of the three pictures is less than 2 percent higher than with individual predictors for each picture. With Netravali's predictor the entropy is sometimes 10 percent higher when the predictor is optimized for the sum of the three pictures.

The advantage of using a good-bad ordering is smaller with the new predictor, the decrease in entropy being about 10 percent. We emphasize however, that the good-bad ordering leads to a nearly monotonically decreasing run-length distribution and therefore the runs are easy to code, while without ordering, the run-length distribution is very irregular. Straightforward run-length codes can be devised which code the pictures obtained after prediction and reordering with an efficiency of about 90 percent.

IV. SUMMARY AND CONCLUSION

We have described a new predictive coding scheme for dithered pictures where the pels used for prediction have variable positions. The prediction is made according to the value of nearby pels that have nearly the same threshold level and according to the value of the threshold of the pel to be coded. We obtain a decrease of 20 to 30 percent in the run-length entropy compared to an earlier prediction scheme in the case of dithered pictures of natural scenes. The bit rate

is reduced to 0.18 to 0.23 bits/pel. We obtain a slightly higher entropy when we fix the position of the pels used for prediction. In the case of graphical pictures, the entropy is about the same as before. The new predictor has the advantage of being nearly independent of the characteristics of individual pictures. In the case of a coding system, the run-length distributions must be used to propose a code whose performances should be near those given by the entropy measures. This prediction scheme was conceived for images dithered with a 4×4 ordered dithered matrix, but can be easily applied to other dither matrices. We envision extensions that would improve the performance in regions that have sharp changes in brightness.

APPENDIX

Determination of the Four Neighbor Pels used for Prediction

The two nearest pels with the same threshold level naturally have coordinates $(i - 4; j)$ and $(i; j - 4)$ since a 4×4 dither matrix is used.

To find the two other pels used for prediction, we examine the structure of the 4×4 ordered dither matrix. Let us number the four 2×2 matrices contained in the 4×4 dither matrix as 0, 1, 2, and 3 according to the increase in their average threshold value. Figure 4 shows their relative positions. We must also remember that the 4×4 matrix is surrounded by identical matrices whose pels can also be used as prediction pels.

For a pel in submatrix 1, the neighboring pel with the next-higher threshold level will be in the same relative position in submatrix 2 with coordinates $(i - 2; j)$. The neighbor with the next-lower threshold level will be in the same relative position in submatrix 0. That neighbor can be either in submatrix 0 of the same 4×4 matrix or in submatrix 0 of the matrix adjacent to the right. Both cases lead to the same distance from the neighbor to the pel to be predicted, but the latter case leads to the smallest correlation with the other prediction pels and is therefore chosen. That prediction pel has position $(i - 2; j + 2)$.

Similarly, if the pel to be predicted is in submatrix 2, the two pels

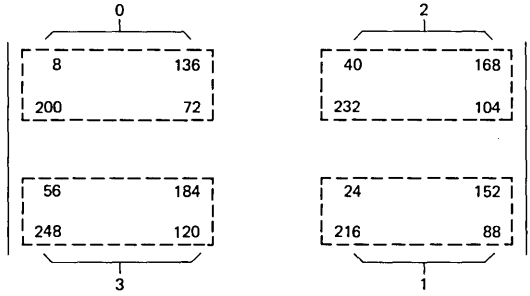


Fig. 4—4×4 dither matrix with the numbering of its 4 submatrices 0, 1, 2, and 3.

with the next threshold level chosen for the prediction have coordinates $(i-2;j)$ and $(i-2,j+2)$.

For a pel in submatrix 3, the pel with the next-lower threshold level will be in the same relative position in submatrix 2. As earlier, there are two possibilities, but correlation comparisons leads us to choose the pel with coordinate $(i-2;j+2)$. The neighbor with the next-higher threshold level will be in submatrix 0, but in each case in a different relative position, as can be verified from Fig. 4. For the pel with threshold level 248, pels with a higher threshold level do not exist. Therefore, the two nearest pels with the next lower-threshold level are chosen as prediction pels.

Similarly, if the pel to be predicted is in submatrix 0, the pel with the next-higher threshold level chosen for prediction has coordinates $(i-2,j+2)$, while the pel with the next-lower threshold level has in each case a different relative position. In this case, when the pel with threshold level 8 is predicted, the two nearest pels with the next-higher threshold level are chosen as prediction pels.

In summary, the positions of the pels used in the prediction are:

1. $i;j-4$,
2. $i-4;j$,
3. $i-2;j+2$,
4. variables.

The position of the prediction pel with variable position is given according to the position of *Sij* within the dither matrix by

$$\begin{vmatrix} i-2;j-2 & i-1;j & i-2;j & i-2;j \\ i-3;j+1 & i-3;j-1 & i-2;j & i-2;j \\ i-1;j+1 & i-1;j-1 & i-2;j & i-2;j \\ i-2;j-2 & i-3;j & i-2;j & i-2\,j \end{vmatrix}$$

REFERENCES

1. J. O. Limb, "Design of Dither Waveforms for Quantized Visual Signals," B.S.T.J., *48,* No. 7 (September 1969), pp. 255–82.
2. B. Lippel and M. Kurland, "The Effect of Dither on Luminance Quantization of Pictures," IEEE Trans. Commun. Technol., *COM-19,* No. 6 (December 1971), pp. 879–88.
3. C. N. Judice, J. F. Jarvis, and W. H. Ninke, "Using Ordered Dither to Display Continuous Tone Pictures on an AC Plasma Panel," Proc. Soc. Inform. Display, *15/4* (Fourth Quarter 1974), pp. 161–4.
4. C. N. Judice, "Date Reduction of Dither Coded Images by Bit Interleaving," Proc. Soc. Inform. Display, *17,* No. 2 (1976), pp. 91–9.
5. J. F. Jarvis, C. N. Judice, and W. H. Ninke, "A Survey of Techniques for the Display of Continuous Tone Pictures on Bi-level Displays," Comput. Graph. Image Process., *5,* No. 1 (March 1976), pp. 13–40.
6. A. N. Netravali, F. W. Mounts, and J. D. Beyer, "Techniques for Coding Dithered Two-Level Pictures," B.S.T.J., *56,* No. 5 (May–June 1977), pp. 809–19.
7. A. N. Netravali, F. W. Mounts, and E. G. Bowen, "Ordering Techniques for Coding of Two-Tone Facsimile Pictures," B.S.T.J., *55,* No. 10 (December 1976), pp. 1539–52.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 3, March 1981
Printed in U.S.A.

An Extension of the CCITT Facsimile Codes for Dithered Pictures

By O. JOHNSEN and A. N. NETRAVALI

(Manuscript received September 24, 1980)

The International Telegraph and Telephone Consultative Committee (CCITT) has recently recommended two redundancy reduction codes for the digital transmission of two-level facsimile. It is of interest to transmit other types of still pictures using these codes. In this paper we develop a method for bilevel dithered pictures, wherein pictures are preprocessed reversibly to permit efficient coding by the CCITT codes. The preprocessing consists of local rearrangement of picture elements to obtain large contiguous black and white areas. The main advantage of these schemes over the existing ones is that they require the addition of a simple processor to a standardized facsimile coder. One of the schemes has the additional advantage that even in the absence of the postprocessor at the receiver, a distorted but recognizable picture is obtained. A compression ratio of 1.5 to 3.5 is obtained with these methods, which is comparable with other coding schemes.

I. INTRODUCTION

The International Telegraph and Telephone Consultative Committee (CCITT) has recently recommended two redundancy reduction codes for the digital transmission of two-level facsimile over the general switched telephone networks.[1] In the future, most facsimile apparatus using redundancy reduction techniques will use these codes. The first code is a one-dimensional run length code called the modified Huffman code (MHC). The second code is a two-dimensional extension of the modified Huffman code, called the modified READ code (MRC).

Dithering is an image processing technique which creates a two-level picture that gives the illusion of a multilevel picture by appropriately controlling the spatial density of black and white picture elements.[2-6] It is useful in systems that use inherently bilevel displays (e.g., plasma panels). Dithering allows a picture to be transmitted as

a two-level picture, thus greatly reducing the number of bits to be transmitted.

The high frequency components of a dithered picture prevent the use of CCITT codes as an efficient redundancy reduction scheme.[5] Judice,[5] Netravali et al.,[7] and Johnsen[8] have proposed several coding techniques for dithered pictures. Judice[5] has proposed a bit interleaving scheme which regroups pels with the same or similar threshold levels, thus allowing the use of run-length coding.

We are interested in coding schemes for dithered pictures that would require only a slight modification to the standardized one-dimensional and two-dimensional CCITT codes. Our aim is to include only a simple preprocessor at the transmitter and a postprocessor at the receiver. The preprocessor would include a dithering processor which transforms the analog signal from the facsimile scanner into the binary data stream corresponding to the dithered image and a precoder which modifies the dithered image in such a way that it can be efficiently coded by the one-dimensional or two-dimensional codes. We only consider reversible transformations. The postprocessor of the receiver must make the inverse transformation. One of the preprocessing schemes that we propose has an interesting property that the ordered picture is visually similar to the original dithered picture. Therefore, a facsimile receiver not containing the postprocessor will be able to reproduce a picture that is in most cases recognizable.

II. ORDERED DITHER

The dithering technique consists of comparing a multilevel image with a position dependent threshold and turning only those picture elements "on" (or "1") where the input signal exceeds the threshold value. The square matrix of threshold values (called the "dither matrix") is periodically repeated over the entire picture to provide the threshold pattern for the whole image.

Several dither matrices can be used. They have been judged on the basis of subjective fidelity of reproduction. A class of dither matrices of special interest are the "ordered dither matrices."[4] A dither matrix of size $n \times n$ simulates $n^2 + 1$ brightness levels where n is a power of 2. Thus, 17 brightness levels can be simulated with a 4×4 ordered dither matrix and 65 brightness levels with a 8×8 matrix. A 4×4 matrix, with 16 threshold levels, is shown in eq. (1),

$$\begin{vmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 5 \end{vmatrix} \tag{1}$$

Figure 1 shows dithered pictures, "Karen," "Engineering drawing," and "House," used for computer simulations. These pictures are 10 cm

Fig. 1a—Dithered test picture of Karen.

by 10 cm and are scanned to generate an array of 512 by 512. The picture intensity is linearly quantized to 8 bits (256 levels) and then dithered. The following dither matrix, obtained from eq. (1) by expanding the threshold levels to cover the entire span of picture intensity, is used:

$$\begin{vmatrix} 8 & 136 & 40 & 168 \\ 200 & 72 & 232 & 104 \\ 56 & 184 & 24 & 152 \\ 248 & 120 & 216 & 88 \end{vmatrix} \tag{2}$$

We mention that the three pictures of Fig. 1 are made from the same originals as used in Refs. 4 and 6, but they are not the same digitized versions.

III. REORDERED PEL CODING

Since the codes to be used are already known, the goal of the ordering is to transform the picture to minimize the coding length. In the one-dimensional case, the runs must be as long as possible. In the

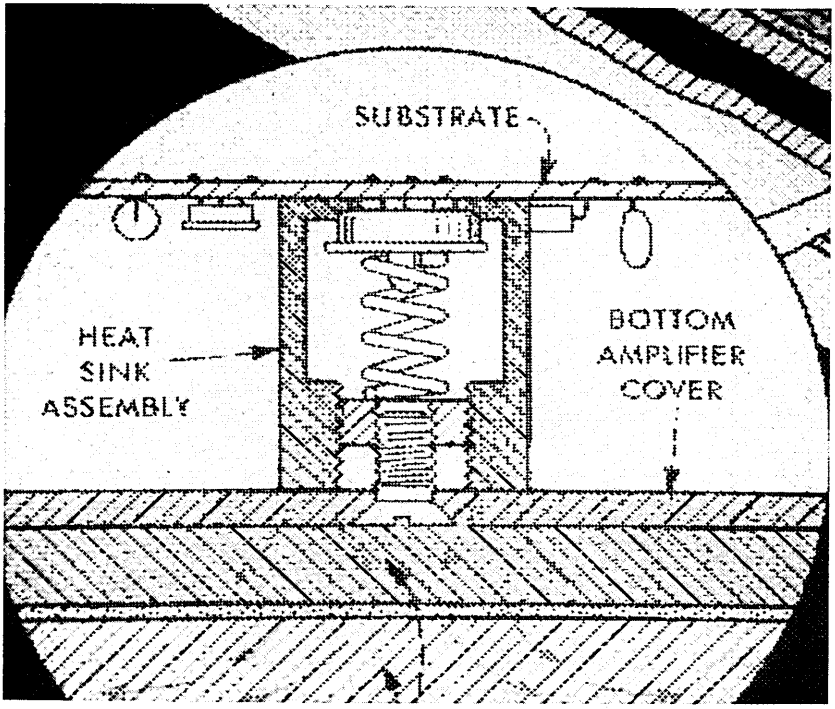


Fig. 1b—Dithered test picture of Engineering drawing.

two-dimensional case, the vertical correlation between runs must also be maximized.

The pel reordering is related to the bit interleaving method proposed by Judice.[5] The main difference is that now the reordering of the pels is done locally. The ordering is made within a few 4 × 4 blocks at a time, typically 1 to 4 blocks. The difference can be seen from the relation between neighboring pels of the ordered pictures. In Judice's schemes, neighboring pels are pels with similar threshold levels of adjacent blocks, while in the local ordering scheme, neighboring pels are pels of nearly the same threshold level of the same or neighboring blocks. Therefore, bit interleaving exploits better the correlation among pels with similar threshold levels, while pel ordering exploits better the correlation among the neighboring pels.

We propose two closely related kinds of pel reordering. One of them is optimized for run-length coding, while the other is optimized for the two-dimensional codes. They have both been conceived for 4 × 4 ordered dither matrices, but in the case of larger dither matrices, simple modifications can be made.

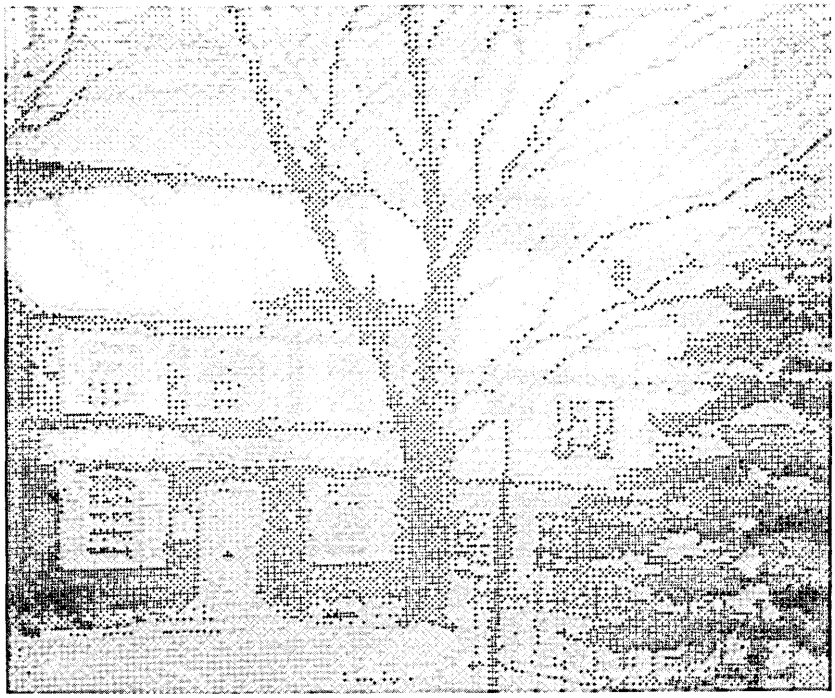


Fig. 1c—Dithered test picture of House.

3.1 Pel reordering for run-length coding

Let us consider the 4 × 4 dither matrix of eq. (1). To have as few runs as possible, the pels with nearly the same threshold levels should be contiguous on the same line. One way to reduce the number of runs, is to reorder the dithered pels for two consecutive blocks slightly differently, for example,

Block A				Block B			
0	1	2	3	3	2	1	0
4	5	6	7	7	6	5	4
8	9	10	11	11	10	9	8
12	13	14	15	15	14	13	12

(3)

Another possibility is to reorder a 4 × 4 block into a 2 × 8 block. Two blocks must therefore be reordered at the same time. For reasons of efficiency, four blocks, $\frac{A}{B}\frac{C}{D}$, are reordered as shown in Table I.

Figures 2 and 3 show the ordered picture of Karen with 4 × 4 and 2 × 8 reordering. It can be seen that these reordered pictures can be

Table I—Reordering of four blocks $\frac{A}{B} \vdots \frac{C}{D}$

A		*B*		*D*		*C*	
0	1	1	0	0	1	1	0
2	3	3	2	2	3	3	2
4	5	5	4	4	5	5	4
6	7	7	6	6	7	7	6
8	9	9	8	8	9	9	8
10	11	11	10	10	11	11	10
12	13	13	12	12	13	13	12
14	15	15	14	14	15	15	14

run-length coded efficiently because there are many long runs. Also note that the reordered pictures are visually degraded versions of the original dithered pictures. The 4 × 4 reordering gives smaller degradation than the 2 × 8 ordering.

3.2 Pel reordering for two-dimensional coding

Pel reordering for two-dimensional coding should increase the vertical correlation between runs, i.e., the reordered picture should have mostly vertical black and white strips. This is accomplished within each block by moving the pels most likely to be white to the left side and the pels most likely to be black to the right side and reversing this

Fig. 2—Karen after 4 × 4 reordering for run-length coding.

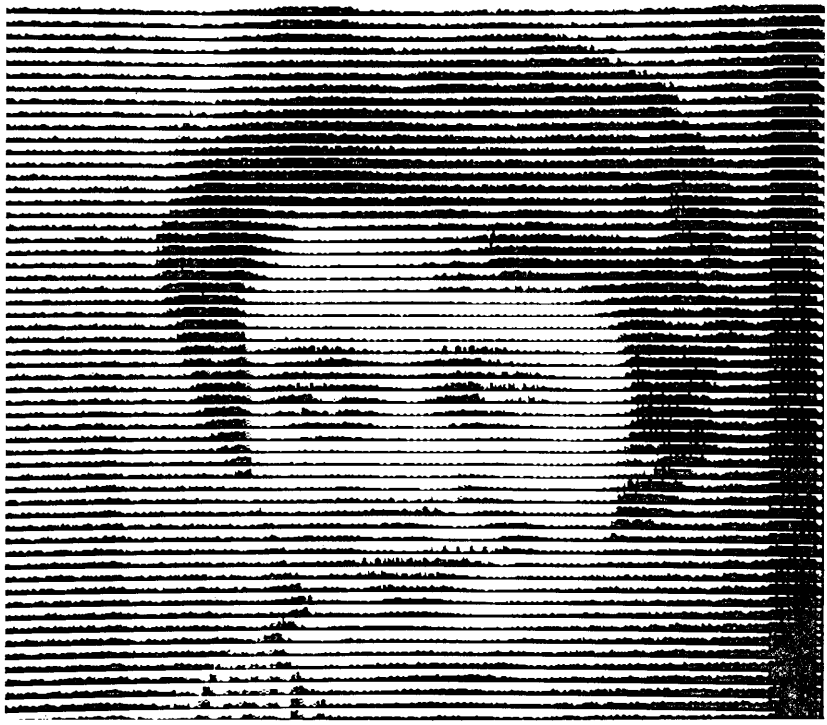


Fig. 3—Karen after 2 × 8 reordering for run-length coding.

process for the next block. The reordering of two blocks, $A \mid B$, would be

Block A				Block B			
0	4	8	12	15	11	7	3
1	5	9	13	14	10	6	2
2	6	10	14	13	9	5	1
3	7	11	15	12	8	4	0

(4)

Further improvements are possible, but the coding efficiency with 4 × 4 reordering is low compared to the other reordering schemes.

Improvement in the coding efficiency is obtained by 8 × 2 reordering. This type of reordering slightly reduces the vertical redundancy between the runs, but that is largely compensated by the reduction in the number of runs. To exploit the maximum amount of correlation, two groups of four blocks of 4 × 4 pels must be considered. These blocks, named A, B, C, D, A', B', C', and D', are shown below:

A	B	C	D
A'	B'	C'	D'

. (5)

Table II—Reordered configuration of eq. (5)

A	0	2	4	6	8	10	12	14	15	13	11	9	7	5	3	1	D
	1	3	5	7	9	11	13	15	14	12	10	8	6	4	2	0	
B	1	3	5	7	9	11	13	15	14	12	10	8	6	4	2	0	C
	0	2	4	6	8	10	12	14	15	13	11	9	7	5	3	1	
B'	0	2	4	6	8	10	12	14	15	13	11	9	7	5	3	1	C'
	1	3	5	7	9	11	13	15	14	12	10	8	6	4	2	0	
A'	1	3	5	7	9	11	13	15	14	12	10	8	6	4	2	0	D'
	0	2	4	6	8	10	12	14	15	13	11	9	7	5	3	1	

Table II shows the reordered configuration.

Another type of reordering is the 16 × 1 reordering. Now the pels are arranged in order of increasing or decreasing threshold level. Eight 4 × 4 blocks shown below,

A	B	C	D	E	F	G	H

,

are transformed into:

$$\begin{array}{lccccccccccl} A & 0 & 1 & \ldots & 14 & 15 & 15 & 14 & \ldots & 1 & 0 & E \\ B & 0 & 1 & \ldots & 14 & 15 & 15 & 14 & \ldots & 1 & 0 & F \\ C & 0 & 1 & \ldots & 14 & 15 & 15 & 14 & \ldots & 1 & 0 & G \\ D & 0 & 1 & \ldots & 14 & 15 & 15 & 14 & \ldots & 1 & 0 & H \end{array} \tag{6}$$

A slight improvement can be obtained by reversing every second group of four lines. Figures 4 and 5 show the picture "Karen" transformed by 8 × 2 and 16 × 1 reordering, respectively.

3.3 Implementation considerations

All the above reordering schemes are simple to implement. Except in the case of 2 × 8 reordering for one-dimensional coding, only four lines with a maximum of 16 pels per line are processed at a time. The basic operation is to address the bits in a different sequence. The reordering should add very little to the cost of the coder and decoder, certainly less than the dithering operation which itself is quite simple.

IV. COMPARISON OF PERFORMANCES

The coding length in bits per pel have been measured for the various reordering schemes and for the bit interleaving schemes of Judice.[3] The three test pictures of Fig. 1 were used. Only the one-dimensional and the two-dimensional codes recommended by the CCITT for standardization[1] are considered. The end of line codeword has been suppressed in the comparisons. In the two-dimensional case, all lines, except the first one, are coded with the two-dimensional code. Note that all the reordering schemes are two-dimensional, even when a one-dimensional code is used afterwards.

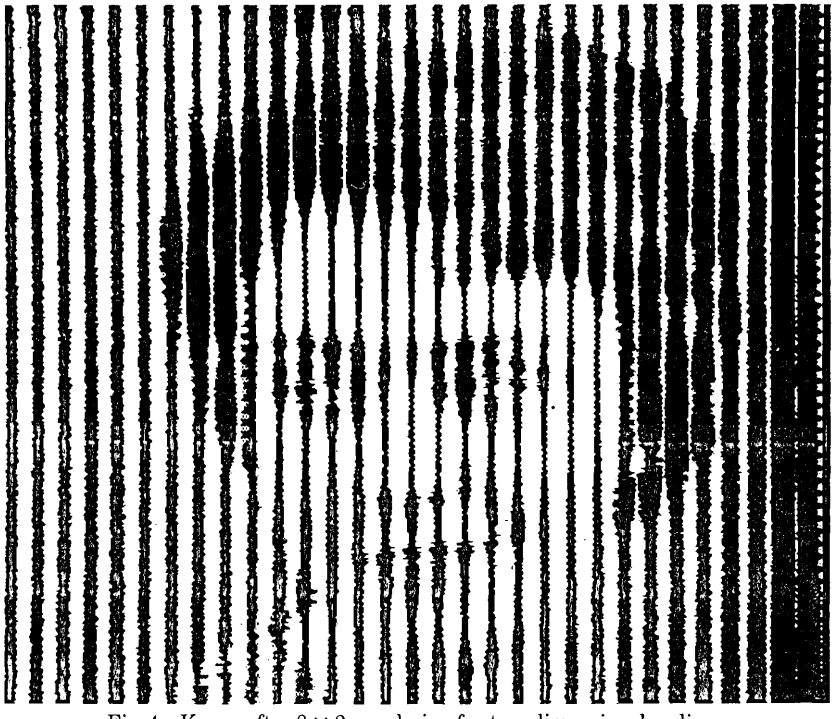


Fig. 4—Karen after 8 × 2 reordering for two-dimensional coding.

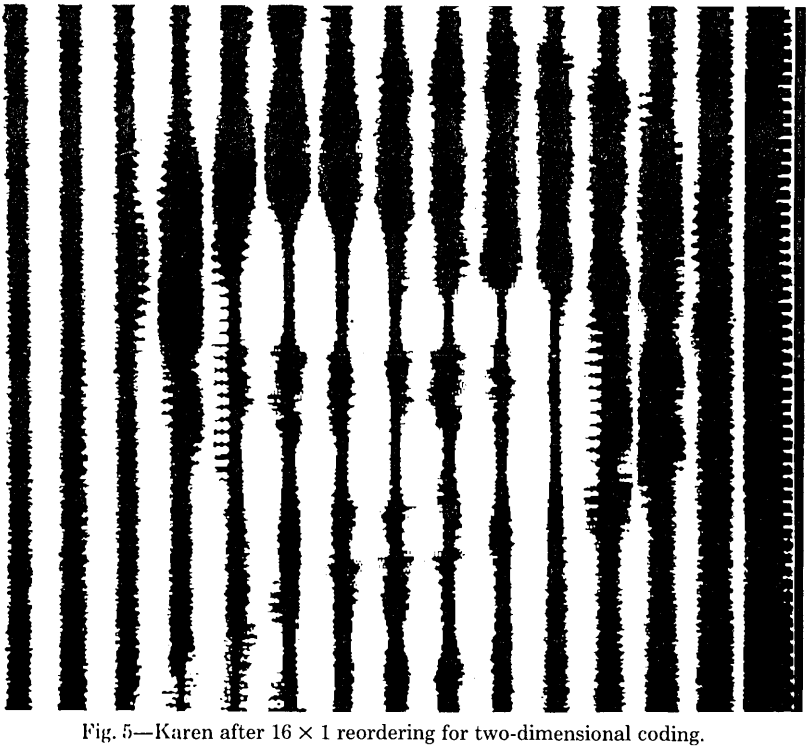


Fig. 5—Karen after 16 × 1 reordering for two-dimensional coding.

Table III—Coding length in bits per pel of the three test pictures when they are reordered and coded with the one-dimensional (1-D) and two-dimensional (2-D) codes proposed by the CCITT for standardization

	Karen		Engineering Drawing		House	
	1-D Code	2-D Code	1-D Code	2-D Code	1-D Code	2-D Code
1-D bit interleaving	0.433		0.607		0.365	
2-D bit interleaving	0.390	0.328	0.592	0.609	0.327	0.291
4 × 4 reordering for run-length coding	0.395		0.661		0.383	
2 × 8 reordering for run-length coding	0.381		0.633		0.372	
4 × 4 reordering for 2-D coding		0.514		0.701		0.475
8 × 2 reordering for 2-D coding		0.321		0.642		0.309
16 × 1 reordering for 2-D coding		0.306		0.621		0.301

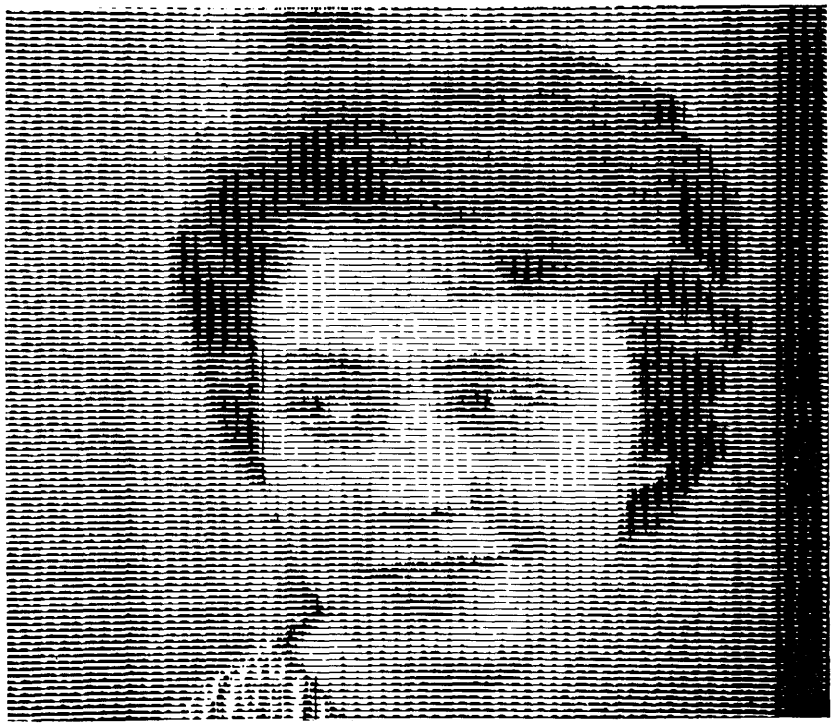


Fig. 6a—Karen after 4 × 4 reordering for run-length coding.

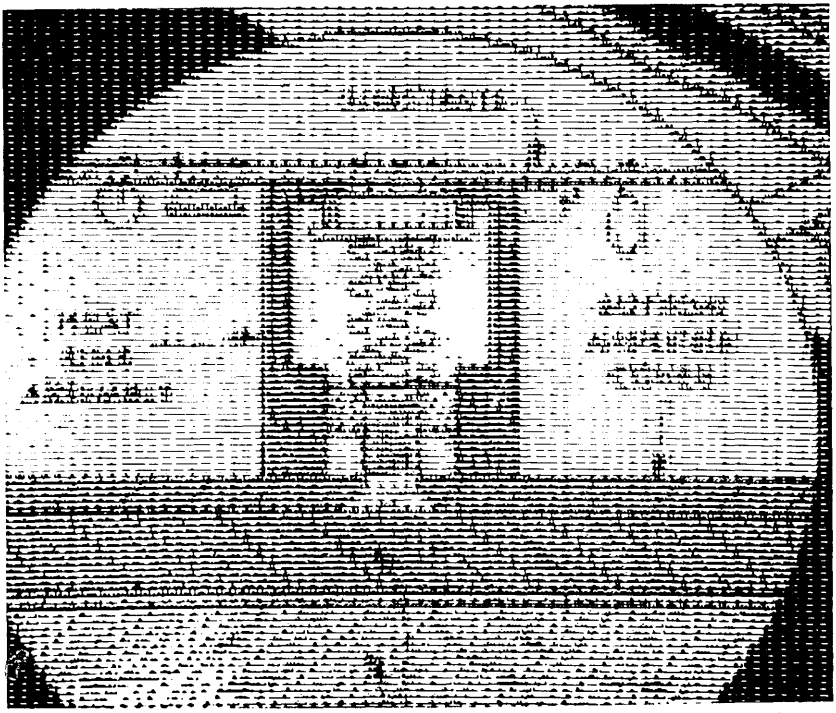


Fig. 6b—Engineering drawing after 4 × 4 reordering for run-length coding.

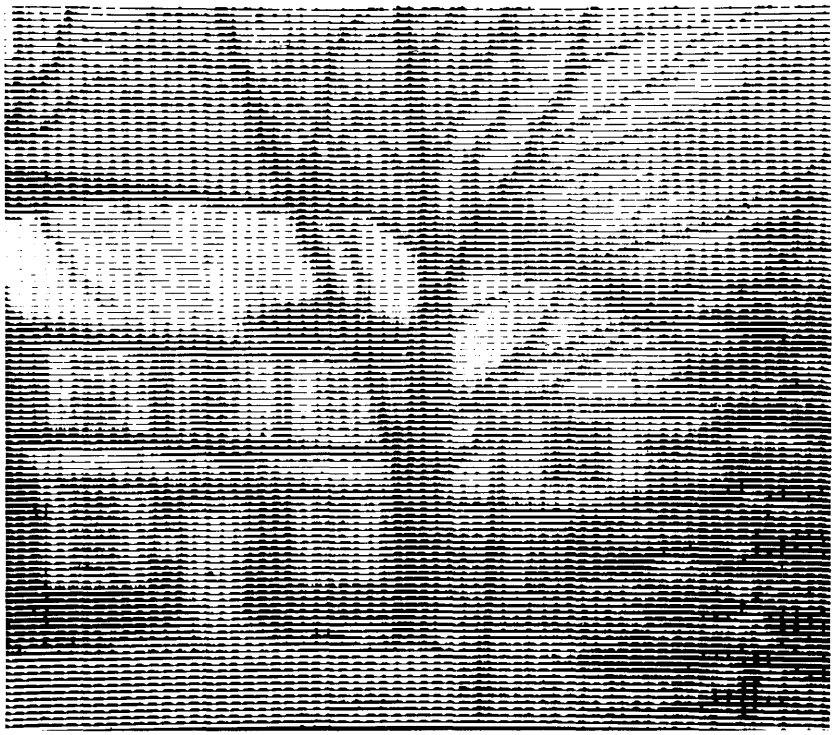


Fig. 6c—House after 4 × 4 reordering for run-length coding.

Table III shows the coding lengths. In the case of the one-dimensional CCITT code, the compressions with the 4 × 4 and 2 × 8 reordering are slightly lower than with the two-dimensional bit-interleaving scheme of Judice. Interest in the 2 × 8 reordering is small because, compared to the simpler 4 × 4 reordering, the decrease in coding length is less than 5 percent.

With the two-dimensional CCITT code, the two-dimensional bit interleaving and the 16 × 1 reordering lead to similar coding lengths. The reordering scheme seems preferable since it is simpler. The two-dimensional bit-interleaving scheme requires either the storage of the entire picture or several scans of the picture. The storage can be reduced to four lines without sacrificing the coding performance, but this requires modifying the CCITT code so that the fourth previous line is used as a reference line rather than the previous line.

One advantage of the 4 × 4 reordering for run-length coding is that the reordered picture is similar to the original dithered picture. Figure 6 shows the reordered versions of the three test pictures. Except for the "Engineering drawing," the other two pictures have reasonable

Table IV—Entropies in bits per pel for the three test pictures when they are reordered

	Karen		Engineering Drawing		House	
	Run-length Entropy	2-D Entropy	Run-length Entropy	2-D Entropy	Run-length Entropy	2-D Entropy
1-D bit interleaving	0.313		0.429		0.276	
2-D bit interleaving	0.293	0.272	0.420	0.478	0.256	0.245
4×4 reordering for run-length coding	0.302		0.468		0.293	
2×8 reordering for run-length coding	0.266		0.433		0.262	
4×4 reordering for 2-D coding		0.436		0.608		0.388
8×2 reordering for 2-D coding		0.271		0.526		0.265
16×1 reordering for 2-D coding		0.247		0.500		0.243

quality. Thus, a facsimile receiver without the dither postprocessor is able to reproduce a recognizable approximation of the dithered picture.

Table IV gives entropy measurements for comparison purposes. The two-dimensional run-length entropy is obtained from the distribution of the symbols used in the two-dimensional CCITT code. The entropies are about 15 to 20 percent lower than the corresponding coding lengths of Table III. Improved codes leading to coding lengths only 10 percent higher than the entropy can be devised, but are outside the scope of this study. Entropy comparisons with Ref. 8 show that higher compression can be obtained by more sophisticated techniques.

V. CONCLUSION

We have shown that dithered pictures can be transmitted efficiently with facsimile apparatus that uses either the one-dimensional or the two-dimensional codes proposed by the CCITT for standardization. The addition of a simple reversible preprocessor at the transmitter and a postprocessor at the receiver is required. The number of bits transmitted is 1.5 to 3.5 times lower than without coding. This compares favorably with compression ratios obtained by other schemes. The preprocessor consists of local reordering of pels. One of the pel reordering schemes has the advantage that a recognizable version of the original dithered picture is obtained with facsimile receivers not containing the postprocessor. This study has been made using only three small dithered pictures and a 4×4 dither matrix. Additional studies using a larger set of pictures and a larger dither matrix are necessary before implementation.

VI. ACKNOWLEDGMENT

The authors thank S. M. Rubin and J. T. Whitted for their help in providing picture processing programs.

REFERENCES

1. R. Hunter and A. H. Robinson, "International Digital Facsimile Standards," Proc. IEEE, *68,* No. 7 (July 1980), pp. 854–67.
2. J. O. Limb, "Design of Dither Waveforms for Quantized Visual Signals," B.S.T.J., *48,* No. 7 (September 1969), pp. 2555–82.
3. B. Lippel and M. Kurland, "The Effect of Dither on Luminance Quantization of Pictures," IEEE Trans. Commun. Technol., *COM-19,* No. 6 (December 1971), pp. 879–88.
4. C. N. Judice, J. F. Jarvis, and W. H. Ninke, "Using Ordered Dither to Display Continuous Tone Pictures on an AC Plasma Panel," Proc. Soc. Inform. Display, *15/4* (Fourth Quarter 1974), pp. 161–4.
5. C. N. Judice, "Data Reduction of Dither Coded Images by Bit Interleaving," Proc. Soc. Inform. Display, *17,* No. 2 (1976), pp. 91–9.
6. J. F. Jarvis, C. N. Judice, and W. H. Ninke, "A Survey of Techniques for the Display of Continuous Tone Pictures on Two-Level Displays," Comput. Graph. Image Process., *5,* No. 1 (March 1976), pp. 13–40.
7. A. N. Netravali, F. W. Mounts, and J. D. Beyer, "Techniques for Coding Dithered Two-Level Pictures," B.S.T.J., *56,* No. 5 (May–June 1977), pp. 809–19.
8. O. Johnsen, "A New Code for Transmission of Ordered Dithered Pictures," B.S.T.J., preceding article in this issue.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 3, March 1981
Printed in U.S.A.

A 200-Hz to 30-MHz Computer-Operated Impedance/Admittance Bridge (COZY)

By L. D. WHITE, R. W. COONS, and R. C. STRUM

(Manuscript received September 12, 1980)

For the past few years the development of ferromagnetic components, particularly for long-haul transmission systems, has relied heavily on large numbers of highly accurate impedance measurements made on a computer-operated impedance/admittance bridge (COZY) developed especially for this work. COZY's accuracy and speed enable a level of component development not otherwise possible. COZY automatically measures complex impedance, temperature coefficients of complex impedance, and disaccommodation factors of ferromagnetic materials, providing accuracies of ±0.05 percent for inductance, ±50 microradian for loss angle, and ±10 parts per million for the small impedance changes associated with determinations of temperature coefficients and disaccommodation factors. COZY is easy to use and makes a measurement in 10 to 20 seconds. Also, the calibration of the bridge unit's capacitance and conductance standards can be checked automatically. Though developed primarily for ferromagnetic component work, COZY is a general-purpose bridge; it measures inductance, capacitance, resistance, and conductance over wide impedance ranges at frequencies between 200 Hz and 30 MHz. This paper describes COZY's hardware, software, and performance.

I. INTRODUCTION

A computer-operated impedance/admittance bridge (COZY) has been developed to have the following features:

- wide frequency range—200 Hz to 30 MHz in 0.01-Hz steps
- wide impedance/admittance range—from a resolution of 0.1 nanohenry for small impedances to a resolution of 0.001 picofarad for small admittances
- high accuracy—high-Q unknowns can be measured to ±0.05 percent for inductance/capacitance and ±50 microradians for loss angle. Changes in impedance/admittance (with temperature,

time, shock, and vibration, etc.) can be measured to ±10 parts per million.

- specifiable signal level—voltage or current may be specified over the nominal range of 0.05 to 5 volts for impedances larger than 100 ohms and 0.5 to 50 milliamperes for smaller impedances. The achieved level is within ±10 percent of the requested level and is measured to ±3 percent.
- relatively fast—20 seconds per measurement
- easy to use yet flexible—the bridge has only a single pair of binding posts to which the unknown is connected and the user needs to specify only a test frequency. However, the user can specify signal levels, frequency runs, and various options for post-processing of the measurement results.
- the options for runs and postprocessing can be changed easily—the software clearly separates the options from the basic measurement process.
- automatic aids for maintaining high accuracy—in particular, the calibration of the bridge unit's standards can be automatically checked.

A microcomputer-controlled environmental chamber with an 18-sample capacity is appliqued to COZY. The combined system provides the following additional features:

- Highly accurate automatic measurements of the changes of the samples' impedances/admittances with environmental conditions—temperatures may be specified to ±0.1° Celsius between −40° and 93° Celsius and relative humidities to ±2.5 percent between 20 and 95 percent, with a minimum dew point of 2.5° Celsius. Soak times, signal levels, multiple frequencies, and environmental runs can also be specified. Average time for a single measurement is 10 seconds.
- Automatic meaurements of the disaccommodation factors (the decrease in permeability with time after demagnetization) of ferromagnetic materials—peak demagnetization currents up to 2 amperes, with a 10-volt maximum, can be specified.

COZY was developed to provide the measurements required in ferromagnetic component development work. Large numbers of highly accurate measurements are required to evaluate materials, structures, and whole components over their operating ranges of frequency, signal level, and environmental conditions and to determine the effects of aging, shock, and vibration.

The most crucial of the measurement requirements that led to the development of COZY were: (1) a basic precision of significantly better than ten parts per million to achieve the desired accuracies in measuring high Q-values and small changes in impedance, and (2) a measurement time much less than a minute to provide the desired quantity of

measurements. The measuring systems that come closest to meeting these requirements are specially developed manual bridges[1] and the 50-Hz to 250-MHz computer-operated transmission measuring system.[2] The manual bridges have satisfactory precision but require many minutes and much care and expertise for a measurement. The computer-operated transmission measuring system, on the other hand, is amply fast but has a basic precision of approximately 100 parts per million.

To meet the objectives for precision requires bridge techniques; pure transmission measurements are not satisfactory. To achieve short measurement times requires automation, and because the amount and complexity of bridge computations are large, the automation has to be done with a computer.

The development of COZY required new design features and measurement procedures. COZY's bridge differs markedly from manual bridges in two ways. First, small impedances are measured with novel bridge configurations based on techniques previously used to calibrate inductance standards.[1,3] Second, all switching of the bridge configurations and setting of the standards is done by a new design of mercury-wetted contact relay that requires very different design considerations than the wafer switches used in manual bridges.

COZY's measurement process differs significantly from measurement processes in manual bridges in three basic areas: selecting the bridge configuration, balancing the bridge, and obtaining the last 1½ decades of the balance. To determine the bridge configuration for a measurement, COZY calculates an approximate value for the unknown from four transmission-type measurements, three of which use predetermined settings of the bridge to provide a calibration of the system. Balancing is done with an iterative process in which capacitance and conductance standards are changed, the ratio of the change in the admittance of the standards to the change in the bridge output is calculated, and the next change to be made in the standards is calculated by multiplying this ratio by the last bridge output. The final 1½ decades of the balance are determined by measuring the bridge output over a one-second period. If the final degree of balance had been limited by noise, this measurement provides increased resolution by noise averaging. On the other hand, if the degree of balance had been limited by the finite size of the smallest steps of the standards, the measurement provides interpolation between these steps.

To provide information suitable for use in the measurement process, the receiver must be phase sensitive and linear right down to zero signal. This is accomplished by using heterodyne techniques to produce two dc signals whose amplitudes, including signs, represent orthogonal components of the bridge's output signal.[4] The accuracy of the representation is one percent.

To achieve the required control of temperature and humidity, the environmental chamber's heaters, compressor, and humidifer water were put under the control of a microcomputer using specially developed firmware. A microcomputer rather than COZY's computer was used to enable COZY's computer to be free for general purpose measurements while the specified environmental conditions and soak times are being achieved.

This paper describes the hardware, software, and performance of the computer-operated bridge and of the facilities added to the bridge to provide temperature coefficient and disaccommodation factor measurements. Section II describes the bridge unit and other basic hardware. The calibration of the bridge is covered in Section III. Section IV describes the software, including the measurement process, interaction with the user, and postprocessing of the measurement results. Section V gives the measurement accuracy and discusses the sources of measurement uncertainty. Section VI concerns the automatic aids for maintaining the accuracy and hardware. The main features of the hardware and software for automatically measuring the disaccommodation factors of ferromagnetic materials and the effects of temperature and humidity on impedance are described in Section VII. Section VIII is a summary.

II. BRIDGE UNIT AND OTHER BASIC HARDWARE

2.1. General

Figure 1 shows the basic hardware blocks for making impedance measurements: a signal generator, bridge unit, voltmeter connected to the bridge unit, receiver, and analog-to-digital converter, all controlled by a computer through an interface and test panel. Figure 2 is a photograph of COZY when put into service. The two and one-half bay cabinet at the left contains from left to right the signal generator, receiver, and bridge unit. Mounted on the horizontal top surface of the half-bay are the bridge's two binding posts to which the unknown is connected. The six-bay cabinet on the right contains the computer and, at the far end, the interface and test panel. In the middle is a teletypewriter. In the background at the end of the six-bay cabinet is a "step-up" unit that provides computer controlled admittance ballast for automatic calibrations of the bridge's admittance standards.

2.2 Bridge unit

2.2.1 Overall

The bridge is a unity ratio type with 100-ohm resistors forming the ratio arms. Four configurations of the other two arms, accomplished by automatic switching, are used to measure the full-admittance range.

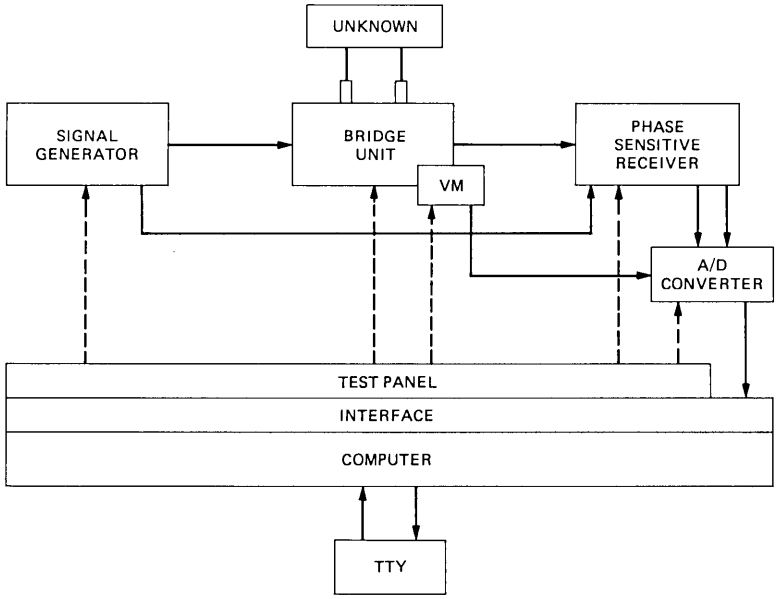


Fig. 1—Block diagram of computer-operated impedance/admittance bridge (COZY).

Figure 3 shows simplified schematics of these configurations. The capacitance standard, C_s, is in the A-D bridge arm for all configurations; the unknown, UNK, may be in either the A-D or the C-D arm; and the conductance standard, G_s, is always in the arm adjacent to the

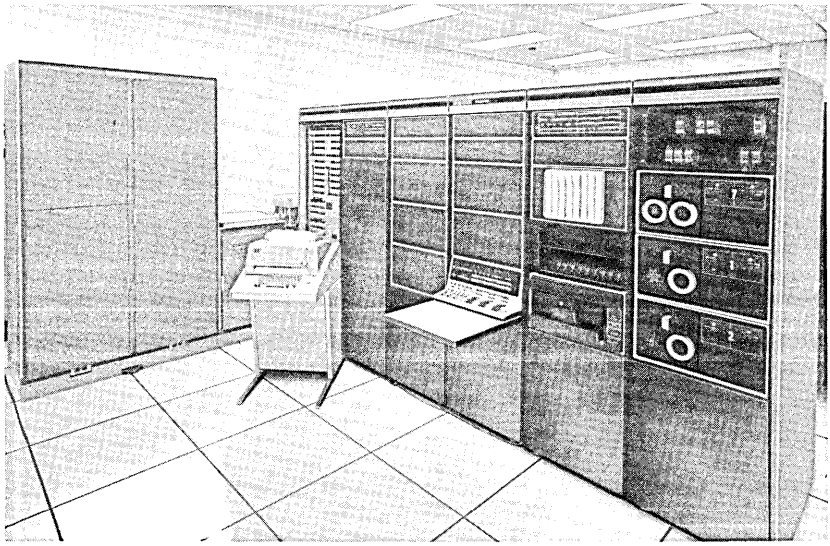


Fig. 2—Computer-operated impedance/admittance bridge (COZY).

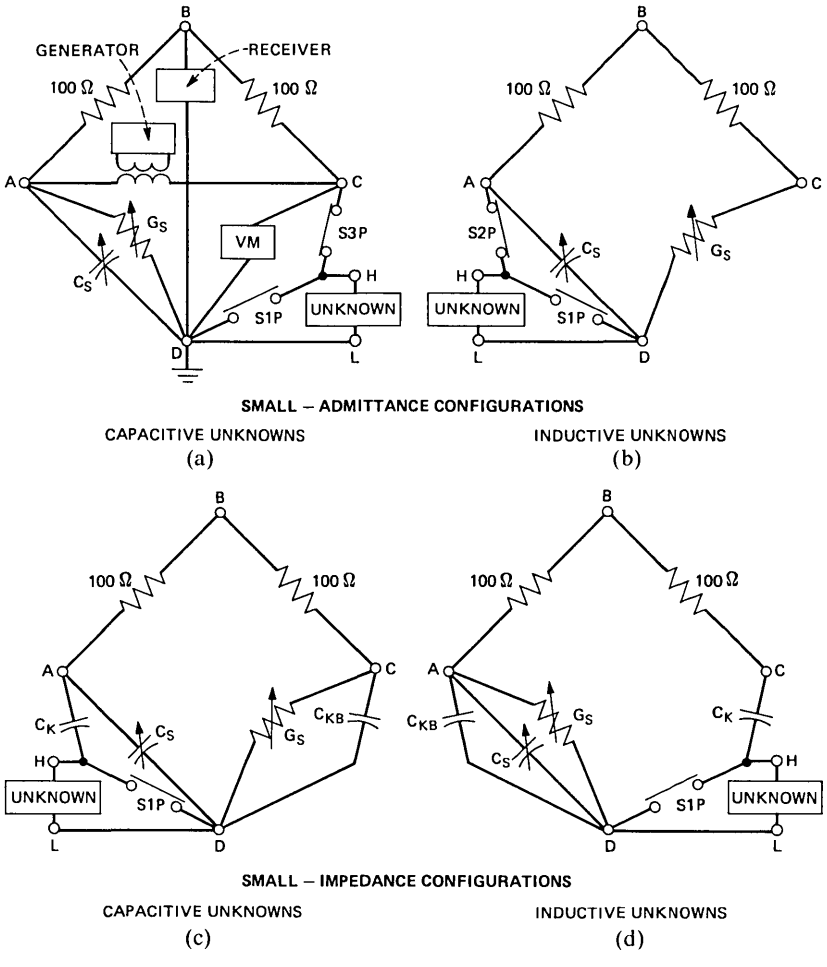


Fig. 3—Bridge configurations.

unknown. The "small-admittance" configuration shown in Fig. 3a is used for capacitive unknowns having susceptances typically smaller than 0.02 siemens and conductances smaller than 0.009 siemens. The Fig. 3b configuration is used for similar-sized inductive unknowns. The "small-impedance" configurations shown in Figs. 3c and 3d are used for capacitive and inductive unknowns, respectively, having susceptances typically larger than 0.02 siemens and/or conductances larger than 0.009 siemens. In these small-impedance configurations measurements are made with one of eleven calibrated capacitors, C_K, in series with the unknown and a similar-sized ballast capacitor, C_{KB}, in the adjacent arm.

As shown in Fig. 3a, signal is applied to the bridge by a transformer

connected between the A and C corners; the receiver is connected between the B and D corners; the voltmeter is connected across the C-D arm; and the D corner is grounded.

Each measurement requires manipulating the capacitance and conqductance standards to balance the bridge twice—an unknown balance with the unknown connected into a bridge arm and a reference balance with the unknown effectively out of the arm. The unknown's admittance is computed from the admittance difference between the two balances. For the small-admittance configurations, shown in Figs. 3a and 3b, the unknown balance is made with the switch in series with the unknown closed and the switch shunting the unknown open (as shown). The reference balance is made with the series switch open and the shuntng switch closed. For the small-impedance configurations, shown in Figs. 3c and 3d, the shunting switch is open for the unknown balance and closed for the reference balance.

The bridge's basic blocks and switches are shown in Fig. 4. Two transformers are required to cover the frequency range. One is used from 200 Hz to 101 kHz and the other, from 101 kHz to 30 MHz. Both are double-shielded and specially developed for bridge use. The transformers' intershield capacitances, 90 pF and 15 pF, are large enough to require the complete disconnection of the unused transformer.

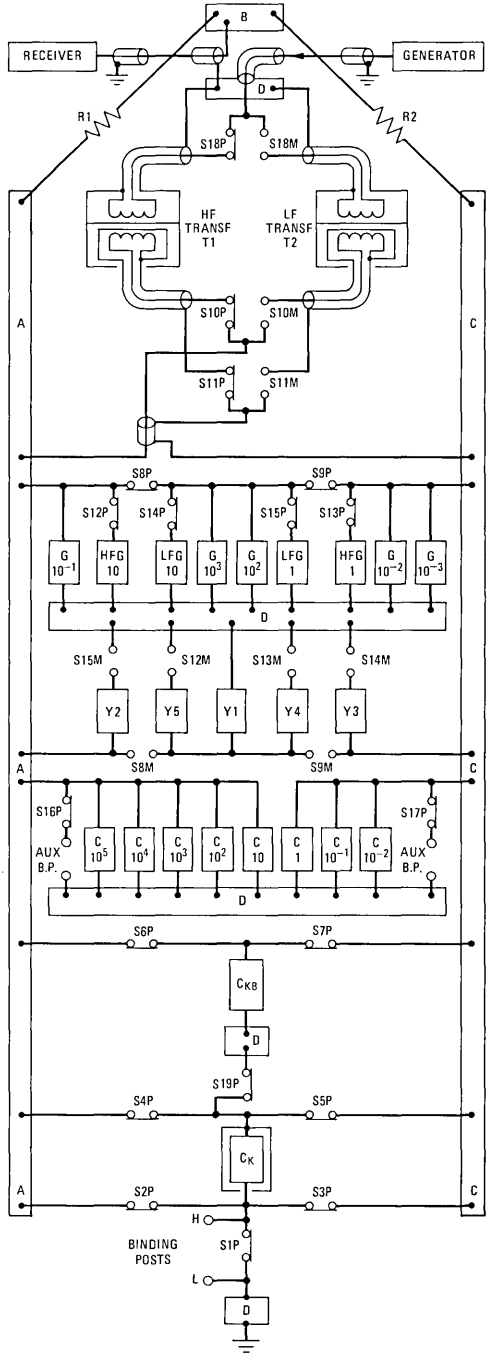
The ratio arm resistors, R_1 and R_2, are 0.01-percent metal film resistors having very small parasitic impedances so that their resistances are frequency independent well beyond the requirements of this bridge. The time constant of the ratio was adjusted to be within 10 ps of zero.

The capacitance standard consists of eight decades covering 1.1 μF in 0.01-pF steps. The 1-, 0.1-, and 0.01-pF per step decades are wired into the C-D bridge arm. Since the capacitance standard is treated as being in the A-D arm, these decades are operated in reverse; that is, their "zero" settings are their maximum capacitance settings.

The conductance standard covers 11,000 μS in 0.001-μS steps and consists of nine decades; low frequency and high frequency versions of the 1- and 10-μS per step decades are necessary to cover the frequency range. The 1000-, and 100-μS and the low frequency 10- and 1-μS per step decades can be switched into either the A-D or C-D arm. The remaining conductance decades are wired into either the A-D or C-D arm.

The distribution of decades among the A-D arm, the C-D arm, and being switched was based on considerations of simultaneously minimizing the total admittance in the arms, the number of leads at a bridge corner, and the frequency dependencies of the decades.

The switched admittances Y_1 through Y_5 compensate for the changes in admittances in the A-D and C-D arms that accompany switching the conductance decades from one arm to the other. For



RESISTORS

R1, R2 – 100 OHM

TRANSFORMERS

T1 – USED 101 kHz TO 30 MHz
T2 – USED 200 Hz TO 101 kHz

CONDUCTANCE STANDARD (9 DECADES)

G 10^{-3} – 0.001 μS/STEP
G 10^{-2} – 0.01 μS/STEP
G 10^{-1} – 0.1 μS/STEP
LFG 1 – 1 μS/STEP (200 Hz – 5 MHz)
HFG 1 – 1 μS/STEP (5 MHz – 30 MHz)
LFG 10 – 10 μS/STEP (200 Hz – 15 MHz)
HFG 10 – 10 μS/STEP (15 MHz – 30 MHz)
G 10^{2} – 100 μS/STEP
G 10^{3} – 1000 μS/STEP

CAPACITANCE STANDARD (8 DECADES)

C 10^{-2} – 0.01 pF/STEP
C 10^{-1} – 0.1 pF/STEP
C 1 – 1 pF/STEP
C 10 – 10 pF/STEP
C 10^{2} – 100 pF/STEP
C 10^{3} – 0.001 μF/STEP
C 10^{4} – 0.01 μF/STEP
C 10^{5} – 0.1 μF/STEP

CONDUCTANCE STANDARD COMPENSATORS

Y1 – COMPENSATES FOR THE RESIDUAL ADMITTANCE OF THE G 10^{2} AND THE G 10^{3} DECADES. WHEN THE DECADES ARE IN THE A-D ARM, Y1 IS IN THE C-D ARM; AND VICE VERSA.

Y2 – COMPENSATES FOR THE RESIDUAL ADMITTANCE OF THE HFG 1 DECADE. WHEN HFG 1 IS BEING USED, IT IS IN THE C-D ARM AND Y2 IS IN THE A-D ARM.

Y3 – COMPENSATES FOR THE RESIDUAL ADMITTANCE OF THE HFG 10 DECADE. WHEN HFG 10 IS BEING USED, IT IS IN THE A-D ARM AND Y3 IS IN THE C-D ARM.

Y4 – COMPENSATES FOR THE RESIDUAL ADMITTANCE OF THE LFG 1 DECADE. WHEN LFG 1 IS BEING USED, IT IS IN THE ARM CONTAINING G 10^{3} AND Y4 IS IN THE ARM CONTAINING Y1.

Y5 – COMPENSATES FOR THE RESIDUAL ADMITTANCE OF THE LFG 10 DECADE. WHEN LFG 10 IS BEING USED, IT IS IN THE ARM CONTAINING G 10^{3} AND Y5 IS IN THE ARM CONTAINING Y1.

SERIES CAPACITANCE STANDARD

C_K CONSISTS OF ELEVEN CALIBRATED CAPACITANCE SETTINGS. THEIR NOMINAL VALUES ARE 10 pF, 30 pF, 100 pF, 300 pF, 1 nF, 3 nF, 10 nF, 30 nF, 100 nF, 300 nF and 1.1 μF. C_K IS CONNECTED IN SERIES WITH THE UNKNOWN WHEN THE UNKNOWN'S ADMITTANCE IS TOO LARGE TO BE MEASURED BY COMPARISON WITH THE CONDUCTANCE AND CAPACITANCE STANDARDS.

SERIES CAPACITANCE BALLAST

C_{KB} CONSISTS OF ELEVEN CAPACITANCE SETTINGS HAVING THE SAME NOMINAL VALUES AS C_K. WHEN C_K IS IN THE A-D ARM, C_{KB} IS IN THE C-D ARM; AND VICE VERSA. C_K AND C_{KB} ARE SET TO THE SAME NOMINAL VALUE AND THE CAPACITANCES OF C_{KB} HAVE BEEN ADJUSTED SO THAT WHEN SWITCH S1P IS CLOSED, THE BRIDGE BALANCES AT LOW SETTINGS OF THE CONDUCTANCE AND CAPACITANCE STANDARDS.

MODE SWITCHES

S1 THROUGH S19 ARE SPECIAL MERCURY-WETTED CONTACT RELAYS WITH TWO MAGNETIC POLES LABELED M AND TWO PLATINUM POLES LABELED P. WITH POWER APPLIED TO A RELAY'S COIL THE MAGNETIC POLES ARE BRIDGED WITH A SHORT CIRCUIT. WITHOUT POWER THE PLATINUM POLES ARE BRIDGED.

D-CORNER

THE FIVE D-SECTIONS REPRESENT THE SINGLE D-CORNER.

Fig. 4—Schematic diagram of bridge unit.

example, Y_1 compensates for the capacitance changes associated with switching the 1000- and 100-μS decades. When the decades are in the A-D arm, Y_1 is in the C-D arm, and vice versa.

The calibrated series capacitor, C_K, and the ballast capacitor, C_{KB}, used in the small-impedance configurations, may be set to: 10, 30, 100, 300, 1000, 3000 pF and 0.01, 0.03, 0.1, 0.3, and 1 μF. The series capacitor, C_K, is contained within a shield and the shield and capacitor can be connected to the A, C, or D corner. The ballast capacitors were adjusted during prove-in so that the bridge balances with low settings on the capacitance and conductance standards when the switch, S1P, across the binding posts is closed.

The bridge was mechanically designed with the following objectives in mind: as low as possible impedances in series with the various components; the components in one arm shielded from the components in the other arms; and the basic blocks in each arm to have independent leads to the junction points with the adjacent arms and the associated generator or receiver connection, thus well defining each bridge corner. The ratio resistors and the A, B, C, and D corner blocks are contained in a central rigid structure. The capacitance and conductance decades are connected to the A or C block and the D block with coaxial cables.

All the circuit components were selected for stability with time, temperature, humidity, and vibration. In addition, the bridge temperature is held constant to better than $\pm 0.05°$ Celsius.

A critical circuit component, developed specially for this bridge, is the relay used for all switching. The relay contains a mercury-wetted contact switch, shown in Fig. 5, with the leads to the normally open contacts made of magnetic alloy and the leads to the normally closed contacts made of platinum. With no power applied to the relay the two platinum leads are shorted by a bar carried by the armature. When power is applied, this bar moves to short the two magnetic alloy leads. The magnetic alloy leads provide part of the magnetic circuit that permits the relay to be operated without external magnets, which would be too bulky. The platinum leads give a stable low-impedance path for the critical circuits; the alloy leads' resistance changes so much with time after a relay is switched, due to temperature changes caused by local heating, that these leads can be used only in very high impedance or noncritical circuits. A specially designed shield and coil assembly electrostatically shields the capsule from its driving coil and surrounds the whole assembly with an electromagnetic shield that completes the magnetic circuit.

The switch is very stable and reproducible. Measurements with manual bridges showed that the switches reset with variations of less than ten micro-ohms in series resistance and one thousandth picofarad in shunt capacitance.

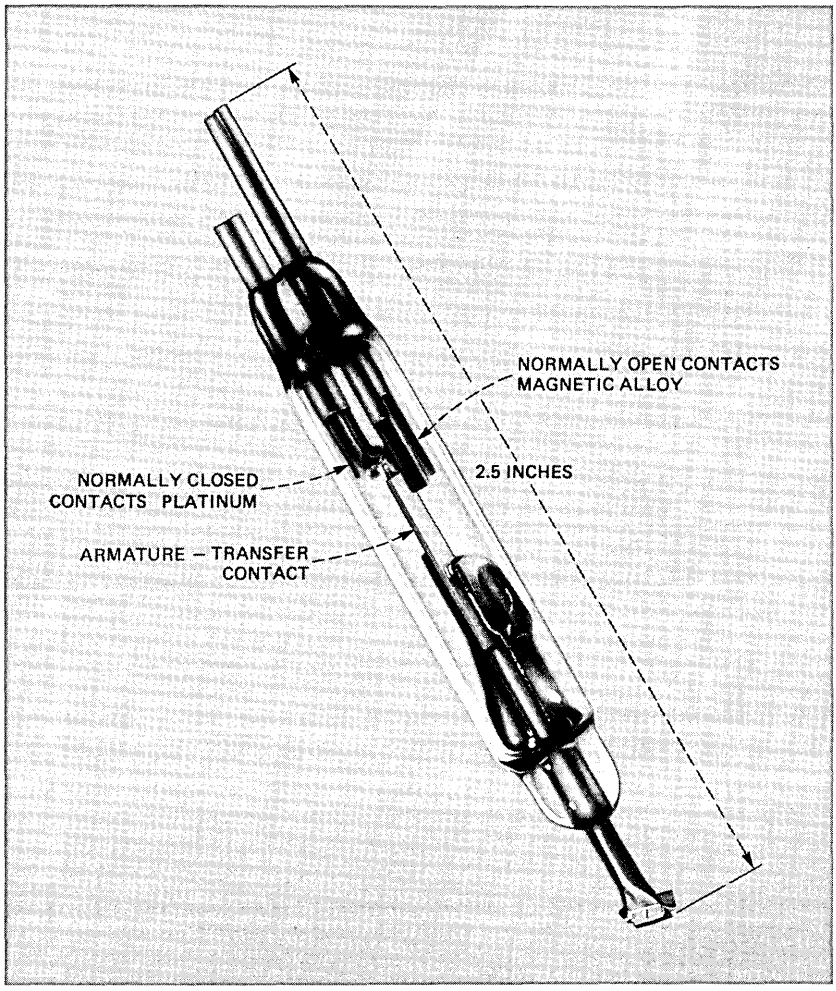


Fig. 5—Special mercury-wetted contact switch used in the bridge unit's relays.

However, the switch has some disadvantages. It must be used upright; it is relatively large with a horizontal center-to-center spacing of three-quarters of an inch; and it has 5-pF capacitance from the armature to the grounded shield.

Figure 6 is a photograph looking down on the bridge. So that the details show, the top cover and the cover of the shield surrounding the ratio resistors have been removed. The junction of the detector lead with this shield is the B corner. The A and C corners are below the high binding post, *H*, which is shown. The D corner is below the A and C corners. The low binding post, *L*, is mounted on the top cover and in use is located above the shield around the ratio resistors.

2.2.2 Design of admittance standards and series capacitor

A fundamental objective in designing the capacitance and conductance standards was to achieve wide frequency performance. The problem is that the admittances of the steps of the decades typically increase with increasing frequency and the percentage increases are larger for the larger steps. As a result, at high frequencies gaps occur in the admittances that can be provided by the standards.

Accommodation for some increases in the admittances of the decades' steps is achieved by using decades that employ 11 settings: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10. The ten-settings provide overlaps at low frequencies in the admittance ranges obtained with successive settings of any decade except the least significant one. For example, at low frequencies the maximum capacitance that can be obtained with the one-setting of the 100-pF per step decade being the most significant setting is 211.10 pF. This capacitance is obtained by setting the 100-pF through 0.01-pF per step decades to 1-10-10.-10-10, respectively. On the other hand, the minimum capacitance obtainable with the two-setting of the 100-pF decade is 200 pF, achieved with decade settings of 2-0-0.-0-0. Thus, there is an 11.1-pF overlap in the capacitance

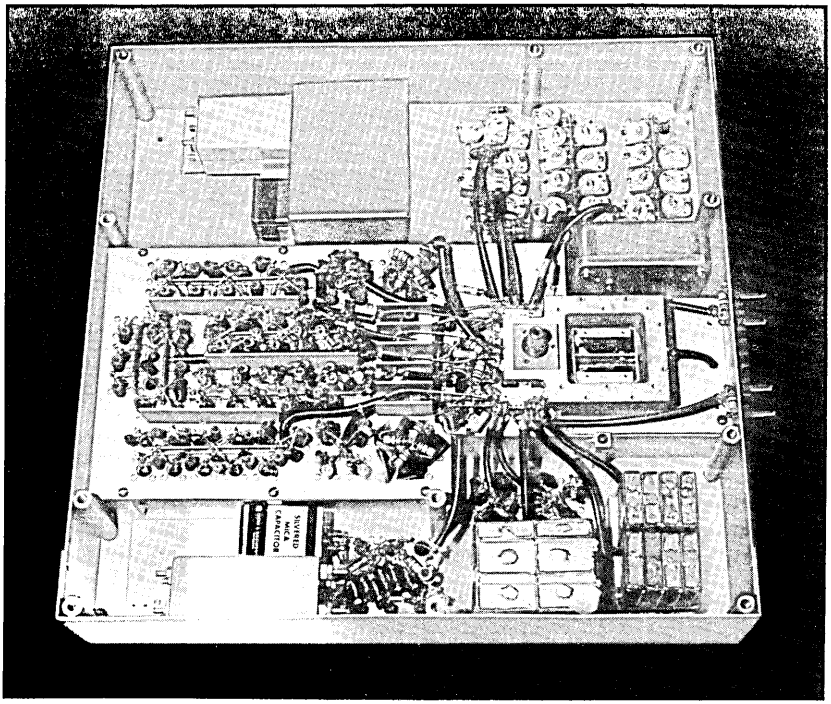
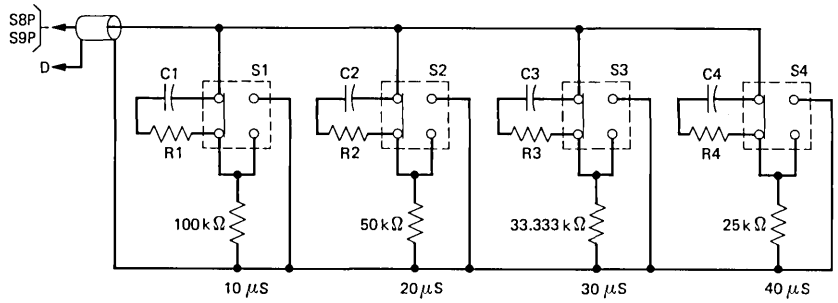


Fig. 6—Top view of bridge unit.



SWITCHES ARE SHOWN UNPOWERED.
PLATINUM CONTACTS ARE CONNECTED TOGETHER.

Fig. 7—Schematic diagram of low-frequency 10-microsiemens per step decade.

ranges provided by the one- and two-settings of the 100-pF decade. Consequently, the capacitance of the two-setting can increase to being 111.1 pF larger than that of the one-setting before a gap develops between their associated capacitance ranges.

Two general types of decades are used in the COZY bridge: an "adding" type consisting of four units that singly and in combination form a complete decade, and a "residual" type consisting of 10 or 11 capacitors (or resistors) switched singly into series with a common much smaller capacitor (or larger resistor). The 100,000-pF, 10,000-pF, 1000-pF, 100-pF, 10-pF, 1000-μS, 100-μS, and the low frequency 10- and 1-μS decades are adding type. The rest of the decades are residual type.

The most complex of the adding type decades is the low-frequency 10-μS per step decade, which is used up to 15 MHz. Figure 7 shows the decade. The individual units are 10, 20, 30, and 40 μS. Capacitors are used to compensate for the 5-pF capacitances between the switch armatures and ground, thereby making the decade's capacitance independent of switch settings. The resistors in series with the capacitors are relatively small and were selected to yield conductances that partially compensate for the increases with frequency of the conductances of the unit resistors as a result of their distributed capacitances. Thus, the admittance differences between the decade's settings are almost pure conductance and as independent of frequency as is practical.

However, at frequencies above 15 MHz a residual type decade is needed to achieve 10-μS steps. The decade contains 11 resistors, ranging from 100 to 404.76 ohms, that can be switched singly into series with a 1500-ohm resistor. Across each resistor is a capacitor adjusted to achieve the same time constant for each resistor and thereby to make the conductance differences between the settings

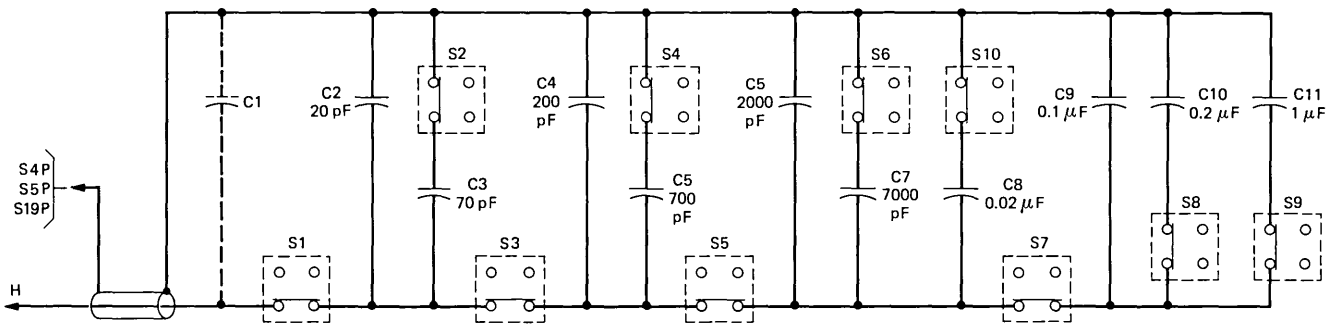
independent of frequency. This decade is not used at low frequencies because it adds 525 μS to the bridge's residual admittance.

As with the bridge standards, a design objective for the series capacitors, C_K, was to minimize the increases in their effective capacitances with increasing frequency. However, because the switches have 5-pF capacitance from armature to ground and 0.7 pF across an open switch, a more difficult objective was to achieve at the smaller settings satisfactorily small values for the capacitances in series with and shunting the binding posts. If these capacitances are too large, gaps will exist in the bridge's impedance coverage at the top frequencies. For example, if the minimum values for these capacitances were 12 pF, then a 1-μH unknown could not be measured at 30 MHz—it could not be measured with a small admittance configuration because its 36-pF resonating capacitance exceeds the capacitance standard's 30-pF range and it could not be measured with a small-impedance configuration because the difference between the unknown and reference balances would be 37 pF.

Figure 8 shows the design of the series capacitor, C_K. The minimum setting is with switch S1 open and switch S2 closed. Switch S2 is closed to put slightly more of the parasitic capacitance across the bridge arm where it is harmless. For this setting, the series capacitance is composed entirely of the capacitance of the wiring to the shield and amounts to 10 pF. The capacitance shunting the unknown is also about 10 pF. These capacitances are small enough to provide continuous impedance coverage between the small-admittance and small-impedance configurations. The 30-pF setting of the series capacitor is obtained by closing switch S1 only, which adds a 20-pF capacitor. The 5-pF armature-to-ground capacitance of switch S1 increases the total capacitance shunting the unknown to 15 pF. The 100-pF setting is obtained by closing switches S1, S2, and S4. However, no additonal capacitance is thrown across the unknown since the armature-to-ground capacitance of switch S2 is across the bridge arm. Switch S4 is closed only to concentrate the remaining parasitic capacitances across the whole bridge arm. This type of switching is continued through the higher values of C_K.

2.3 Other basic hardware

The generator contains a frequency synthesizer as the signal source and can supply between 1 millivolt and 10 volts in 0.05-dB steps to a 75-ohm load. The criteria used in selecting the synthesizer included low phase noise and low harmonics, and in selecting amplifiers, high linearity. These criteria are important because many bridge balances are narrow band. In these balances phase noise contributes to the noise at balance and so must be kept low. Also, harmonics are not



NOMINAL VALUE OF SERIES CAPACITOR	CLOSED SWITCHES (CLOSED SWITCHES ARE NOT POWERED)
10 pF	S2
30 pF	S1
100 pF	S1, S2, S4
300 pF	S1, S2, S3
1,000 pF	S1, S2, S3, S4
3,000 pF	S1, S2, S3, S4, S5
0.01 μF	S1, S2, S3, S4, S5, S6
0.03 μF	S1, S2, S3, S4, S5, S6, S10
0.1 μF	S1, S3, S5, S7
0.3 μF	S1, S3, S5, S7, S8
1.1 μF	S1, S3, S5, S7, S9

SWITCHES ARE SHOWN UNPOWERED.
PLATINUM CONTACTS ARE CONNECTED.

C2 THROUGH C10 WERE ADJUSTED SO THAT THE SERIES CAPACITORS WERE WITHIN ONE PERCENT OF NOMINAL.

Fig. 8—Schematic diagram of the series capacitor, C_K.

strongly attenuated and thus care must be taken to prevent them from intermodulating in the receiver to produce a false fundamental. Harmonics in the generator's output are more than 30 dB down from the fundamental.

The receiver's minimum detectable signal is no larger than 0.05 microvolts (with a one-second measurement time) and the receiver is linear for input signals up to at least 5 volts. Maximum voltage gain is one million and at full gain intermodulation of 0.16-volt second and third harmonics (corresponding to 30 dB down from a 5-volt fundamental) produces less than two microvolts of fundamental referred to the input. Crosstalk corresponds to less than 0.05 microvolts of fundamental at the input. Settling times and overload recovery times are no more than 1 ms except for input frequencies below 4.9 kHz, where the settling time is 17 ms.

Between 120 kHz and 30 MHz, two stages of frequency translation are used. The first stage uses a synthesizer as a local oscillator and produces an IF signal at 28 kHz. The second stage uses two ring modulators in parallel to translate the IF signal to two dc voltages. The local oscillator sources for the modulators are two 28-kHz sine waves, with a 90 degree phase difference between them, generated from the same signal sources used to generate the IF signal.

Between 200 Hz and 120 kHz, three stages of frequency translation are used: from 200 Hz to 4.9 kHz, the 28-kHz IF is preceded by a 97-kHz IF; and from 4.9 kHz to 120 kHz, by a 528-kHz IF.

A low-pass filter preceding the first mixer attenuates harmonics of the test frequency so that they do not generate significant IF signals by intermodulation in the mixer. The receiver contains 29 of these low-pass filters and each filter is used over a frequency range of approximately two-thirds of an octave, i.e., the filters cover 200 to 315 Hz, 315 to 480 Hz, etc. The filters below 45 kHz are active and have input impedances of 600 $\pm$ 2 ohms. The filters above 45 kHz are passive and have 75-ohm input impedances and return losses larger than 15 dB.

The receiver's two dc outputs are read one after the other into the computer by the A/D converter. Noise averaging is done by taking successive pairs of readings with 1.4 ms between each pair. The A/D converter covers the range from -10 Vdc to $+10$ Vdc with a 15-bit (including sign) output. At the converter's input is a 48-channel multiplexer.

A specially developed voltmeter is connected across one arm of the bridge and puts out a dc voltage proportional to the rf voltage at its input. Four gain settings are used to cover the 0.05- to 5-volt range and the ratios of the dc to rf voltages are within two percent of the nominal values. One percent accuracy is achieved with simple computer corrections. Voltages below 0.05 volts can be measured with

some degradation in accuracy. The response time for one percent accuracy is 24 ms. An important feature of the voltmeter is that its input admittance is very stable and small; it is 7.2 pF and 0.4 μS at 100 kHz.

The test panel contains a light for observing and a switch for manually controlling the state of each computer-controlled relay. Manual control of the relays is valuable for prove-in and trouble diagnosis of the system. Observation of the lights during the measurement process yields information on whether the process is going well, and if not, where the source of trouble is.

The computer originally included automatic priority interrupt, 24K words of core memory, a 256K-word fixed head disk and three magnetic tape drives. Since then the computer has been upgraded to include 32K words of core memory, two 1¼ M-word cartridge disks, and floating-point hardware.

III. CALIBRATION OF THE BRIDGE UNIT

3.1 General

Calibration values are determined for the capacitance and conductance of each step of the capacitance standard, the conductance standard, and the series capacitors and of the various admittances that appear in the equivalent circuits used in data reduction. These determinations are made at the 18 frequencies of 0.2, 0.5, 1, 2 kHz, $\cdots$, 2, 5, 10, 15, 20, and 30 mHz.

The equivalent circuits for data reduction are given in Section 3.2 and the basic calibration procedures are given in Sections 3.3 and 3.4. However, note that in some cases the actual calibration values assigned are based on more than a single method of determination.

3.2 Equivalent circuits for data reduction

Data reduction is based on simplified equivalent circuits relating an unknown's admittance to the admittance difference between the unknown and reference balances. An objective in choosing the equivalent circuits was that their various admittances could be evaluated with relatively simple procedures and available external admittance standards.

Figure 9 shows the equivalent circuits used for the unknown and reference balances for the small-admittance configuration when measuring capacitive unknowns. Only three admittances per balance are necessary to characterize the circuitry connecting the unknown to the bridge corners. These equivalent admittances may, of course, be strongly frequency dependent. In the reference balance, the impedance of the short-circuiting switch across the binding posts is so small compared to the impedance in series with it and the impedance of the

unknown that the unknown's impedance does not significantly affect the reference balance. Thus, the three equivalent admittances for the reference balance reduce to one.

From Fig. 9, the admittance difference, $Y_U - Y_R$, between the unknown and reference balances is given by

$$Y_U - Y_R = \frac{Y_X + Y_{HD}}{1 + Z_{CH}(Y_X + Y_{HD})} + Y_{CD} - Y_{CR}, \qquad (1)$$

where Y_X is the admittance of the unknown. As part of the calibration procedure, the admittance difference, Y_{UR}, between unknown and reference balances with nothing across the binding posts, is measured

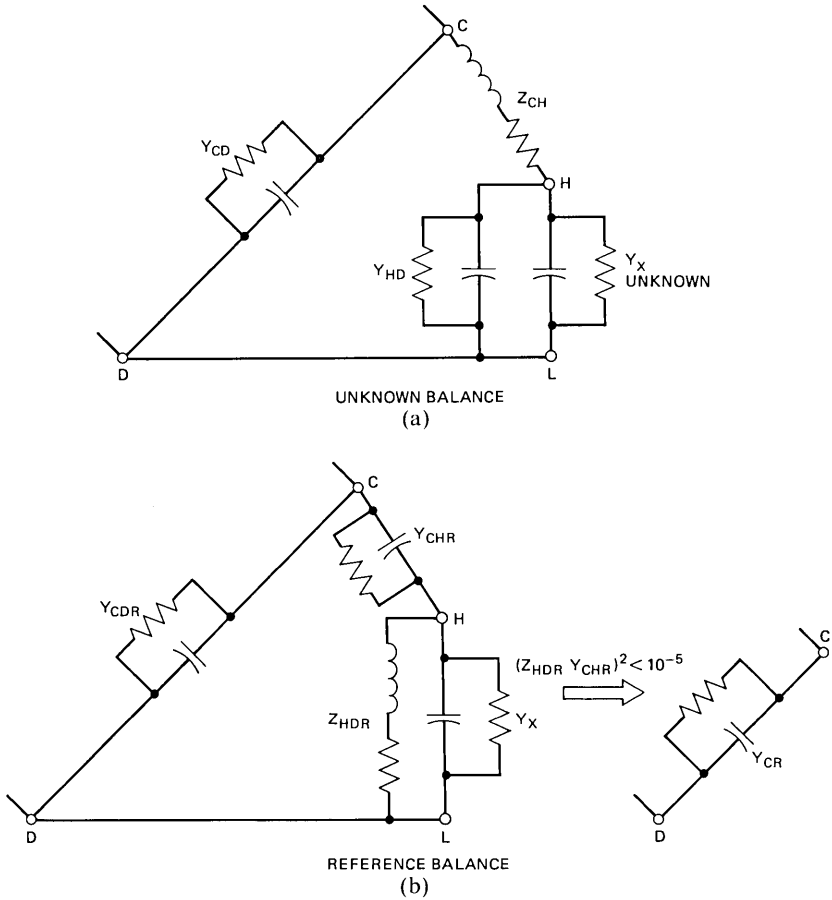


Fig. 9—Equivalent circuits used for reducing measurement data taken with the small-admittance configuration of Fig. 3a.

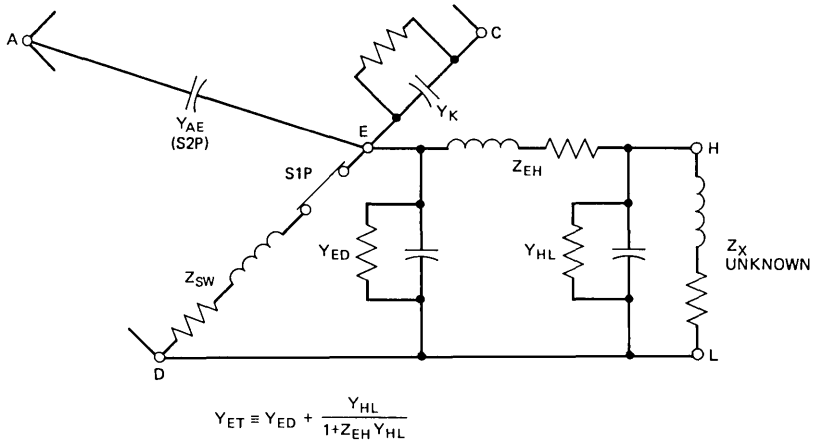


Fig. 10—Equivalent circuit used for reducing measurement data taken with the small-impedance configuration of Fig. 3d.

and stored. This difference is given by

$$Y_{UR} = \frac{Y_{HD}}{1 + Z_{CH}\, Y_{HD}} + Y_{CD} - Y_{CR}. \tag{2}$$

Substituting the expression for $Y_{CD} - Y_{CR}$, obtained from eq. (2) into eq. (1), yields

$$Y_U - Y_R = \frac{Y_X}{[1 + Z_{CH}(Y_X + Y_{HD})]\cdot[1 + Z_{CH}\, Y_{HD}]} + Y_{UR}. \tag{3}$$

This equation is the basis for data reduction. Values for Y_{HD} and Z_{CH} are determined and stored in files as part of the calibration process.

For unknowns having small inductive admittances the values for the equivalent circuit admittances are for the A-D arm.

Figure 10 shows the equivalent circuit for data reduction of inductive unknowns measured with the small-impedance configuration. The unknown balance is made with the short-circuiting switch, S1P, open; and the reference balance, with the switch closed. Y_K is the effective admittance between the C corner and the E junction. Y_{AE} is the admittance across the open switch, S2P (see Fig. 4), to the A corner and is about 0.7 pF. The admittance difference, $Y_U - Y_R$, between the settings of the standards for the unknown and reference balances is

$$Y_U - Y_R = \frac{-(Y_K^2 - Y_{AE}^2)}{[1 + Z_{SW}(Y_K + Y_{AE} + Y_{XE})](Y_K + Y_{AE} + Y_{XE})}, \tag{4}$$

where

$$Y_{XE} = Y_{ED} + \frac{Y_X + Y_{HL}}{1 + Z_{EH}(Y_X + Y_{HL})}. \tag{5}$$

The equivalent circuit for measuring small impedance capacitive unknowns differs from Fig. 10 only in that the unknown is connected into the A-D arm.

3.3 *Calibration of the bridge standards and of the parasitic admittances for the small-admittance configurations*

Determining the calibrations for the capacitance and conductance standards involved four very different phases: (1) the individual steps of the standards were intercompared and consistent calibrations for the standards were obtained by a step-up calibration[5,6]; (2) the magnitude and phase of the bridge ratio were determined; (3) the admittance, Y_{HD}, shunting the binding posts in the small-admittance configuration for measuring capacitors was evaluated; and (4) the impedance, Z_{CH}, in series with the binding posts and absolute values for the capacitance and conductance standards were determined from measurements of external standards.

In a step-up calibration, each incremental step of each decade is compared with the full range of the next smaller decade. For example, consider the calibration of the steps of the 10-pF per step decade. The bridge is set in the configuration for measuring small capacitance, Fig. 4a, and external adjustable capacitance and conductance ballast is connected to the bridge's binding posts. The capacitance standard's 10-pF through 0.01-pF decades are set to 0-10.-5-5; the conductance standard is set to 0-0-5-5.-5-5-5; and the bridge is balanced by adjusting the external ballast. Then the 10-pF decade is set to one, the 1-pF decade is set to zero, and the bridge is rebalanced using the 0.1-pF decade, the 0.01-pF decade, and the conductance standard. Assume the standards' settings at this second balance are 1-0.-5-2 pF and 0-0-5-5.-5-1-2 μS. From the bridge settings we would calculate that the first step of the 10-pF decade is 0.03 pF and 0.043 μS larger than the full 10-pF range of the 1-pF decade. Similarly, the size of the step between the one- and two-settings of the 10-pF decade is then compared with the full range of the 1-pF decade by making balances with these decades set first at 1-10 and then at 2-0. The steps of the 1-pF decade are in turn compared with the full range of the 0.1-pF decade and the steps of that decade, with the 0.01-pF decade.

The 0.01-pF decade cannot be compared with a smaller decade—there is none. Instead, the individual incremental steps of the decade are intercompared via transmission-type measurements made with the receiver, and the results are processed to yield a calibration for the one- through ten-settings relative to the full range of the decade.

After all the steps of all the decades have been intercompared, the data is processed to yield a consistent calibration. The processing starts at the smallest decades and proceeds up both the capacitance

and conductance decades. The calibration calculated for the conductance decades is applied in reducing the data for the capacitance decades and vice versa.

The step-up calibration for the capacitance standard differs from the absolute calibration in magnitude and phase angle, the same percentage difference and the same phase angle difference for each setting of each decade. Thus, the capacitance and conductance corrections to be added to the step-up calibrations for the individual steps are proportional to the capacitances of the steps. Similarly for the conductance standard, the differences between its step-up calibration and its absolute calibration are a percentage for the conductance of each step and a capacitance proportional to the step's conductance for the capacitance of each step.

The magnitude and phase angle of the bridge ratio were determined using the auxiliary binding posts symmetrically connected across the A-D and C-D bridge arms via mode switches S16P and S17P (see Fig. 4). Bridge balances were made with: (1) nothing across the auxiliary binding posts; (2) a capacitance C_A connected to the A-D binding posts, and a similar sized capacitance C_C connected to the C-D binding posts; and (3) C_C connected to the A-D binding posts and C_A to the C-D binding posts. The effects of any differences between the series impedances of the internal coaxial leads between the bridge corners and the auxiliary binding posts were calculated and taken into account by making similar balances using a different capacitance value for C_A and C_C.

The admittance, Y_{HD}, shunting the binding posts in the small-admittance configuration was measured by using switches S3P and S2P to connect Y_{HD} first into the C-D arm and then into the A-D arm. Y_{HD} is one-half the difference between the two resulting balances, corrected for the effects of the impedances in series with Y_{HD} and of the armature-to-ground capacitances in switches S2P and S3P.

The impedance, Z_{CH}, in series with the binding posts was determined and absolute calibrations for the bridge standards were obtained by measuring external standards. Two external capacitance standards were measured in terms of the step-up calibration. The data-reduction equations with known values for the unknowns and the admittance Y_{HD} were then solved for the value for the series impedance Z_{CH} and the percentage and phase-angle corrections to be made in the step-up calibration for the bridge's capacitance standard. Then, an external conductance standard was measured to determine the percentage and phase angle corrections to be made in the step-up calibration of the bridge's conductance standard.

The external capacitance standards used at frequencies up to 100 kHz were mica capacitors calibrated on a Type 12 capacitance bridge[7] that has a basic accuracy of ±50 ppm for capacitance and ±25 micro-

radians for loss angle. At frequencies above 100 kHz, the external capacitance standards were parallel plate capacitors with air dielectric and geometries permitting their conductances and the frequency dependencies of their capacitances to be calculated. The low-frequency capacitances of these standards were obtained by measuring them on the bridge at 100 kHz using the calibration obtained with the external mica capacitance standards. The parallel plate capacitors were not disturbed between the 100-kHz measurements and the higher-frequency measurements.

The external conductance standards were 100- and 1000-ohm metal film resistors whose conductances and shunt capacitances were determined by various procedures using the Type 12 capacitance bridge and a Wheatstone bridge having a basic accuracy of ±10 parts per million. The conductances and capacitances of these resistors are not significant functions of frequency.

The impedance in series with the binding posts when they are connected into the A-D arm was evaluated by comparing measurements made in the A-D arm with measurements made in the C-D arm. Also, the admittance difference between the unknown and reference balances with nothing connected to the binding posts was measured with the binding posts in the A-D arm.

Step-up calibrations of the standards, including the intercomparisons of the steps and the data reduction, are done automatically at frequencies up to 5 MHz. For these calibrations the adjustable capacitance and conductance ballast connected to the bridge's binding posts is a specially developed "step-up unit" under computer control. This unit contains capacitance and conductance decades similar to those in the bridge. The steps of the step-up unit's decades are calibrated by automatically measuring them with the bridge.

Above 5 MHz the performance of the step-up unit is unsatisfactory and special manually adjustable capacitance and conductance ballasts are used. The intercomparison process is semiautomatic, with the operator adjusting the ballast and the computer balancing the bridge, recording the results, and reducing the data.

Measurements of the admittance differences between the unknown and reference balances with nothing connected to the bridge's binding posts are also done automatically.

3.4 Calibration of the series capacitors and parasitic admittances for the small-impedance configurations

The admittances/impedances in the equivalent circuit of Fig. 10 for the small-impedance configurations were evaluated in four phases.

The direct capacitance, C_{AE}, from the E junction to the A corner via the open switch S2P was measured at 10 kHz with the Type 12 capacitance bridge.

The admittances, Y_K and Y_{ET}, in series with and shunting the unknown were determined from various bridge balances. For example, at the setting for minimum series capacitance three balances were made: one with the two admittances in series and one each with the individual admittances connected across the C-D arm. Corrections for the shunt admittances and series impedances of the switches used to connect in the individual admittances enter into the determinations.

The admittance, Y_{HL}, across the binding posts was estimated to be 2 pF.

The switch impedance, Z_{SW}, and the binding post impedance, Z_{EH}, were determined from measurements of the impedance differences between the individual impedances and the impedances in parallel. Three bridge balances were made using the small-impedance configuration and the largest series capacitor calibrated at the frequency of the determination. One balance was with the switch, S1P, closed and nothing across the binding posts. Another balance was with the switch open and the binding posts short-circuited by a metallic plate having effectively zero impedance. The third balance was with the switch closed and the binding posts short-circuited.

The bridge construction is so symmetrical that the values for the circuit elements for the unknown in the A-D arm are the same as for the unknown in the C-D arm.

The measurements and data reduction are done automatically for the series and shunt admittances, Y_K and Y_{ET}, and for the switch and binding post impedances, Z_{SW} and Z_{EH}.

IV. SOFTWARE AND MEASUREMENT PROCESS

4.1 Software structure

The structure of the software for making measurements embodies a clearly defined division of responsibility between the measurement function and the user interaction and post processing functions. This enables the measurement center personnel responsible for the day-to-day operation to respond to changing customer needs without getting involved in the measuring process itself.

Three modules, labeled INPUT, OUTPUT, and CNTROL, provide for the user interaction and post processing. They were written by the measurement center. One module, MEASUR, provides the measurement process and was written by the developers of COZY. The INPUT module conducts the dialogue with the user. When the dialogue is completed, INPUT passes the collected data and program control to the CNTROL module. CNTROL oversees the making of measurements according to the user-supplied data. For each individual measurement, i.e., a single unknown at a single frequency and a single signal level, CNTROL passes the measurement frequency and signal level to MEASUR. MEASUR then

makes the measurement. When all the specified measurements have been made, CNTROL passes the data and program control to the OUTPUT module. OUTPUT processes the measurement results according to the user requests specified to INPUT and presents the processed data in the desired formats.

The INPUT, CNTROL, and OUTPUT modules are each separate overlays. Part of MEASUR is in the CNTROL overlay but the bulk of MEASUR is in two additional overlays.

4.2 MEASUR—The measurement module

The module MEASUR contains all the routines required to measure one unknown at one frequency and one signal level. The test frequency and the desired signal level are passed to MEASUR via dedicated locations in the COMMON storage area. Similarly, MEASUR fills other dedicated locations in COMMON with the measurement results. Included in these results are: the unknown's admittance as seen at the bridge binding posts, the measured signal level applied to the unknown, and information specifying the bridge configuration used.

Figure 11 shows a flowchart for MEASUR. MEASUR starts by tuning the generator and receiver for the specified test frequency and setting the signal level and the receiver gain to their minimum values. Then the calibration data for the capacitance standard, the conductance standard, the series capacitors, and the bridge's parasitic admittances are obtained. For measurement frequencies that correspond to calibration frequencies, the data is read from disk files. For measurement frequencies between the calibration frequencies, admittance values are obtained by interpolation between the values at the calibration frequencies next above and below the measurement frequency. The interpolation formulas are based on the physical causes for the frequency dependencies. The formulas were checked by comparing calibration values at a calibration frequency with values obtained by interpolation using the next lower and higher calibration frequencies.

To efficiently select a bridge configuration and start the balancing process, an approximate value for the unknown's admittance is determined. Preparatory to determining the approximate value, the signal level is set so that during the determination the level applied to the unknown will not exceed that specified for the actual measurement. This level is set by effectively placing the voltmeter across the A-C bridge diagonal, i.e., across the secondary of the transformer, and adjusting the generator to the voltage requested by the user. For this preliminary level setting, a specified current is expressed as the corresponding voltage across a 100-ohm resistance, which is approximately the impedance in series with the unknown during the determination.

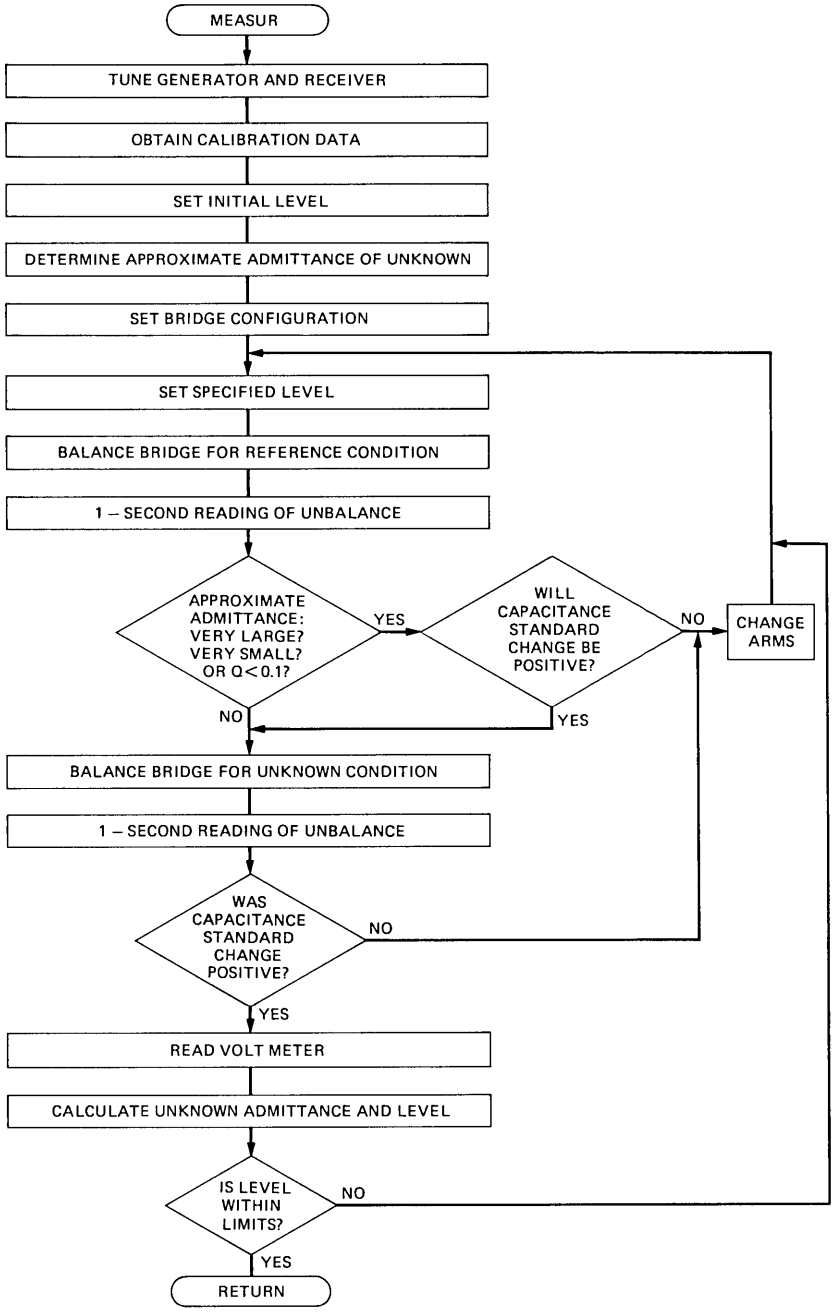


Fig. 11—Flowchart for the software module, MEASUR, that makes a measurement at a single frequency and signal level.

The approximate admittance of the unknown is calculated from measurements of the bridge output for four separate conditions, three of which serve to calibrate the system. All four conditions use the conductance standard in the C-D arm set to 9900 μS, which is approximately the admittance level between the small-impedance and small-admittance configurations. The three conditions used to calibrate the system are: a "short," an "open," and 6280 μS of susceptance in the A-D arm. The admittance values used for the "short" and "open" are based on the parasitic admittances of the switches, S1P and S2P, used to achieve the "short" and "open." The 6280-μS susceptance is obtained by an appropriate setting of the capacitance standard, and the actual susceptance and conductance of the setting are obtained from the standard's calibration data. The fourth condition uses the unknown in the A-D arm.

A small-admittance configuration is used if the conductance of the unknown is less than 90 percent of the 10,000-μS range of the conductance standard and if the magnitude of the unknown's susceptance is within 0.02 siemens and 75 percent of the range of the capacitance standard. Otherwise a small-impedance configuration is used.

When a small-impedance configuration is selected, the program chooses the largest series capacitor that is calibrated at the test frequency and that meets both of the following requirements: (1) the impedance of the capacitor and unknown in series must be greater than 20 ohms, and (2) the capacitance and conductance changes between the series capacitor alone and the series combination of the capacitor and the unknown must be within 75 percent of the range of the capacitance standard and 90 percent of the range of the conductance standard.

The arm into which the unknown is connected is determined by the sign of the unknown's susceptance.

After the bridge is set to the selected configuration and the bridge standards are set at values calculated to yield a bridge balance, the signal generator is adjusted to apply the requested level to the unknown. The level computations are based on the approximate value for the unknown and on the bridge configuration.

The bridge is then balanced at the reference condition. In the balancing process, the bridge's capacitance and conductance standards are iteratively adjusted toward balance, and the receiver gain is iteratively increased to maintain the receiver's output within the working range of the A/D converter. The changes to be made in the bridge standards are computed from the bridge's output, which is approximately proportional to the admittance difference between the A-D and C-D bridge arms. The constant of proportionality, in terms of siemens-per-volt, is computed after each change in the standards. This

computation has to include the relative gains of the receiver before and after changing the standards. At frequencies above 45 kHz, the impedances of the filters following the rf attenuators are so poorly known (15 dB return loss) that the insertion loss of the last rf attenuator must be determined at the time of its being removed. This determination is made by adjusting the bridge unbalance to yield a suitably sized signal and reading the receiver's output with the attenuator in and then with it out.

When the bridge is very nearly balanced, the bridge's output signal is small and the resulting signal-to-noise ratio may be too small for accurate determinations of the siemens-per-volt. Accordingly, the siemens-per-volt is specifically calibrated for a final time when the bridge unbalance is within 0.5 percent of the total admittance across the C-D bridge arm. Between this degree of unbalance and complete balance, the siemens-per-volt will stay constant to within 0.25 percent.

The iterative process used to converge to balance is terminated when one of three conditions has been met: (1) the unbalance is within the smallest steps of the standards plus an allowance for noise; (2) the unbalance is within the rms deviation due to noise; or (3) there have been 50 iterations (a trouble condition).

The admittance of the remaining unbalance is calculated from a one-second reading of the receiver's outputs and the final siemens-per-volt calibration. So receiver's output signal is large enough to be read accurately and/or the output noise is well above the resolution of the A/D converter, the mean of the higher-valued receiver output is made greater than 5 volts or the standard deviation of the noisier receiver output is made greater than 0.2 volts. Then 700 pairs of samples of the outputs are taken over a one-second period.

A check is made of the correctness of the choice of bridge arm if the approximate admittance of the unknown differs by a factor of 100 or more from the 9900-μS admittance level used in determining the approximate value. For such admittances the determination of the approximate susceptance is not made with sufficient accuracy to assure consistently correct arm selection. Similarly the choice of arm is checked if the Q-value is less than 0.1 because the determination of the unknown's susceptance is not made accurately enough in the presence of relatively large conductance.

The check of the arm selection is made by leaving the bridge standards set at the reference balance; setting the S1P, S2P, and S3P switches for the unknown balance; and reading the receiver's output. If the susceptance change indicated by the output would require decreasing the setting of the capacitance standard to achieve balance, the unknown is switched into the other bridge arm and the setting of the signal level and the making of the reference balance are repeated.

When the arm selection is satisfactory after the one-second reading is complete, an unknown balance and one-second reading are made.

Then a second check on arm selection is made. If the capacitance of the unknown balance and its one-second reading is less than that for the reference balance, the arm selection is changed and the measurement is repeated starting at setting the signal level.

When the arm selection is proper, the voltage across the C-D arm of the bridge is read while the bridge is still set at the unknown balance. The admittance of the unknown and the signal level being applied to it are calculated. If the level is within 10 percent of the specified value, the measurement is complete and MEASUR returns. Otherwise, the signal level is reset and the measurement is repeated. However, this time the value of the unknown is known more accurately than the first time so the achieved signal level will be closer to the specified level.

4.3 User interaction and postprocessing

The simplest use of COZY consists of connecting the unknown to the bridge and entering a test frequency via teletypewriter dialogue with the program. After approximately 20 seconds, the measured value, the measurement uncertainty, and the signal level are typed out. In measurements where the signal level is not specified, as in this example, four volts are applied to the bridge.

Up to 15 frequencies and a signal level may be specified in a run. The signal level may be specified in terms of voltage across the unknown or current through it. Signal levels below the nominal minima of 50 millivolts or 0.5 milliamperes can be specified, but the measurement precision may be degraded by receiver noise. Above 15 MHz, the maximum signal levels decrease from 5 volts or 50 milliamperes to 0.4 volts or 4 milliamperes at 30 MHz.

The postprocessing options include having the results expressed in terms of a parallel model (e.g., parallel capacitance and conductance), a series model, and magnitude and angle. Also, Q-values may be requested. In addition, an inductor can be measured at several frequencies, and the change in effective inductance with frequency can be represented as a capacitance across the inductor.

When a cable or fixture is used to attach a component to the bridge, one or two measurements (at each frequency in the case of a frequency run) may be used to characterize the cable. Then, after the component has been measured, the effects of the connecting cable or fixture are automatically corrected for. The characterization measurements are of the open-circuit admittance and/or the short-circuit impedance of the cable or fixture. The circuit model for the cable or fixture is that of a uniform transmission line.

The teletypewriter, a line printer and/or magnetic tape may be specified as the output medium.

V. ACCURACY

5.1 Accuracy for measuring the impedance/admittance of components

Figure 12 shows on a reactance chart contours of ±0.05-, ±0.25- and ±1-percent uncertainties in the measurements of the inductance or capacitance of components having large Q-values. Figure 13 shows contours for ±50- and ±250-microradian uncertainties in measuring the loss angles of such components. These loss angle uncertainties

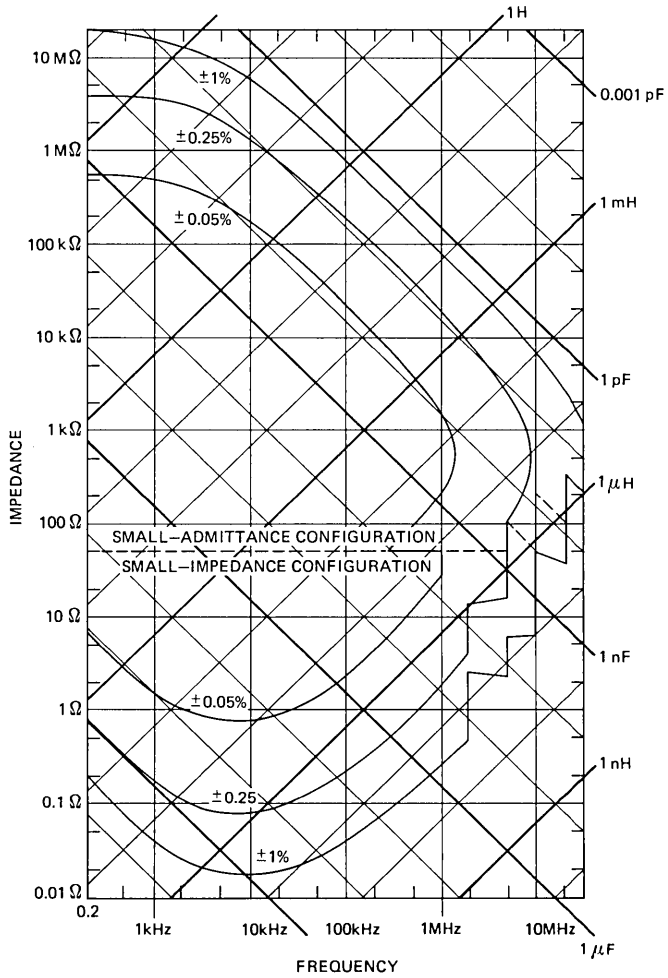


Fig. 12—Inductance/capacitance measurement uncertainty for high-Q components.

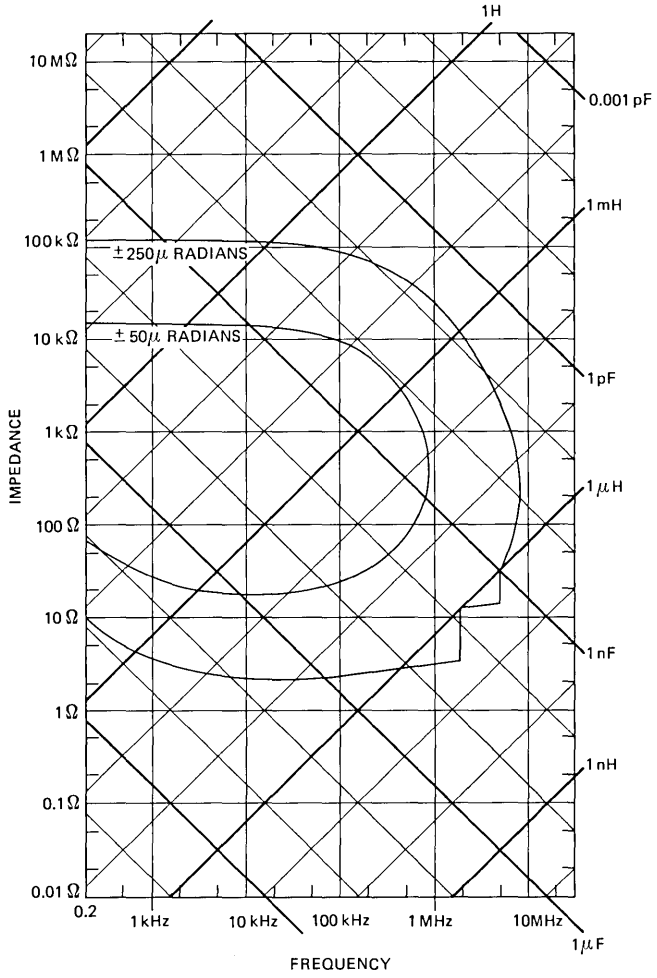


Fig. 13—Loss angle measurement uncertainty for high-Q components.

correspond to ±5 percent uncertainty in measuring Q-values of 1000 and 200. Figure 14 shows ±0.05-, ±0.1-, and ±1-percent uncertainty contours for measuring the resistance or conductance of components having small Q-values. Figure 15 shows contours of ±100-, ±1000-, and ±10,000-microradian uncertainties in measuring the phase angle of these components. For clarity all four figures show smoothed contours for the small-impedance configurations. The exact contours would have small sawtooth wiggles as successively smaller series capacitors were used with increasing frequency. Also small differences between the uncertainties for measuring inductors and capacitors have been ignored.

The confidence factor for the uncertainty contours is 75 percent. That is, at a contour there is a 75 percent probability that the error in a measurement is less than the uncertainty associated with the contour. Well within a contour the confidence factor is much higher than 75 percent. For example, well within the ±0.05 percent contour of Fig. 12 the uncertainty approaches ±0.02 percent and the probability that the error is within 0.05 percent approaches unity.

The uncertainties shown in the figures are for voltages across the C-D bridge arm between 0.05 V and 5 V. For small-admittance components, these voltages correspond to test voltages across the component

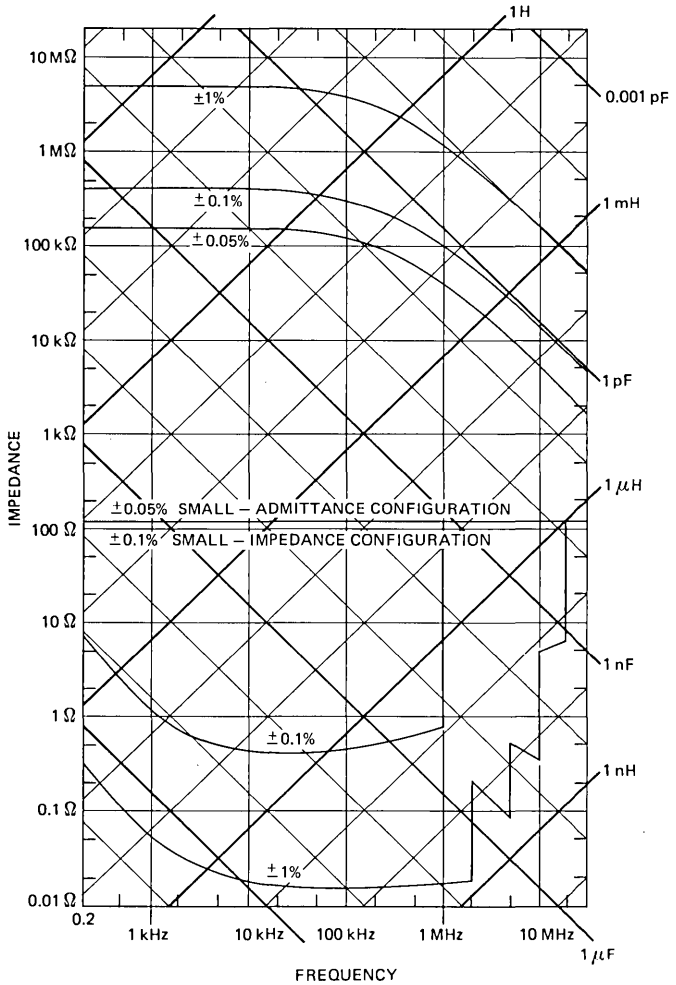


Fig. 14—Resistance/conductance measurement uncertainty for low-Q components.

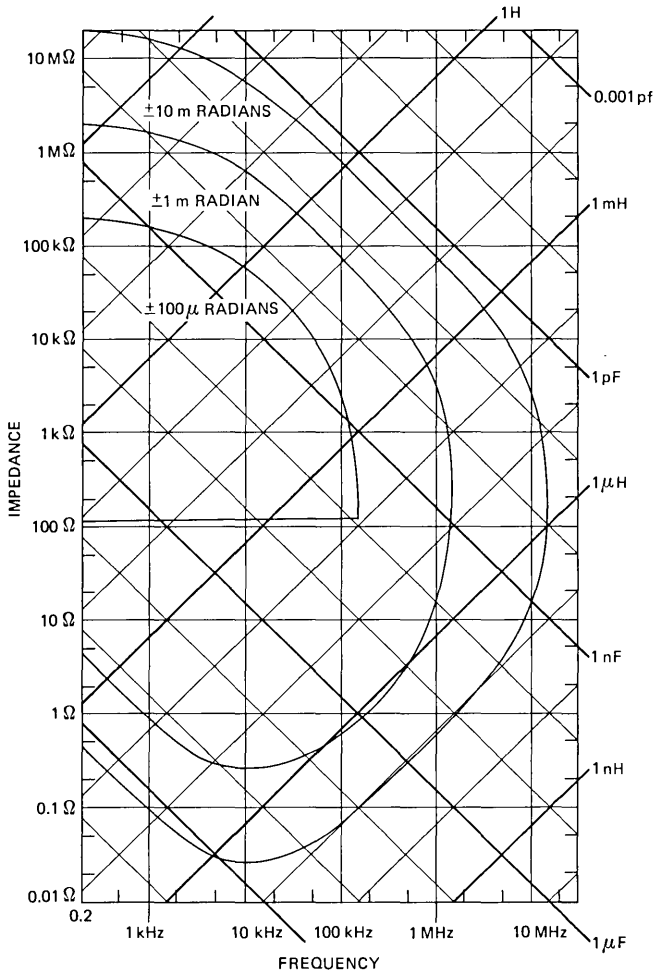


Fig. 15—Phase angle measurement uncertainty for low-Q components.

between 0.05 V and 5 V; for small-impedance components, to test currents through the components of approximately 0.5 mA to 50 mA.

For components measured with the small-admittance configurations, the basic measured quantity is the admittance difference between a measurement of the component plus connecting leads and a measurement of the open-circuited leads. This admittance difference can be translated from COZY's binding posts to the other end of the connecting leads by correcting for the effect of the leads' series impedance. This impedance is obtained by measuring the leads with a short circuit in place of the component. Figures 12 through 15 apply to the admittance difference as seen either at COZY's binding posts or at the component's end of the connecting leads.

For components measured with the small-impedance configurations, the uncertainty contours are for the impedance difference between a measurement of the component at the end of connecting leads having an inductance of 0.1 μH or larger and a measurement of a short-circuit at the end of the leads. This requirement for a minimum 0.1-μH inductance in the connecting leads arises because the condition used for the reference balance differs from that for the unknown balance by the closing of the switch, S1P in Fig. 4, across the binding posts. If the impedance connected to the binding posts is small compared to that of the switch, then the impedance difference between the unknown balance and the reference balance is insensitive to the impedance connected to the binding posts. Since the inductance of the switch is 0.025 μH, using connecting leads having inductances of 0.1 μH or more provides good sensitivity for measuring the inductance of the leads with the component connected to them and the inductance of the leads with the short-circuit connected to them. In addition, with 0.1-μH leads, errors in the calibration of the impedance of the switch have approximately equal effects on the measurements of the leads with the component connected and of the leads with the short-circuit connected. As a result, errors in the calibration of the switch's impedance have only slight effects on the impedance difference between the measurements.

For components measured with the small-impedance configurations, the uncertainty contours also apply to the impedance difference after being translated to the component's end of the connecting leads by correcting the above impedance difference for the connecting leads' shunt admittance, which is obtained by measuring the leads open-circuited.

Figures 12 through 15 were obtained from an analysis of the effects on measurements of the uncertainties in the individual quantities entering into the measurement results. The uncertainties for these quantities were arrived at by a combination of: (1) analysis of the uncertainties in the determinations of the individual quantities by the calibration processes; (2) intercomparisons of the measurements of the same unknown made in different ways (e.g., made with both the small-admittance and small-impedance bridge configurations); (3) measurements of some standards whose admittances are known by other means (e.g., specially constructed parallel plate capacitors and the inductance standards of Ref. 1); and (4) much use of the bridge for the past eight years.

5.2 Sources of uncertainty in small-admittance configuration measurements

The sources of uncertainty for measuring a component's capacitance or inductance using the small-admittance configuration are in the

capacitance calibrations of the bridge standards and in the inductance calibration for the impedance, Z_{CH} in Fig. 9, in series with the binding posts. The uncertainty in the calibration of the capacitance of the capacitance standard at frequencies below 100 kHz is ±0.02 percent. This value represents the uncertainties due to aging and to transferring a calibration from the Type 12 capacitance bridge. The basic uncertainty of the Type 12 bridge is ±0.005 percent. Checks of the COZY bridge show its variations with time to be less than ±0.01 percent. At frequencies above 4 kHz the end term uncertainty in the capacitance calibration for large capacitances is ±0.02 pF and results from: mutual capacitances between the standard's decades; accumulation in the calibration values for the larger steps of the small uncertainties in each of the many balances that contribute to their calibrations; and there being several capacitance and conductance decades involved in any one balance setting. At frequencies below 4 kHz the end term uncertainty corresponds to a susceptance resolution of ±0.5 nS.

An uncertainty of ±5 nH is assigned to the calibration of the parasitic series impedance, Z_{CH}. However, it is not known how much of this comes from the uncertainty in the value for Z_{CH} and how much comes from mutual inductances between the decades of the capacitance standard.

The 0.02-pF and 5-nH uncertainties produce uncertainties in the measurements of the external standards used to determine absolute calibrations for the bridge's capacitance standard. The minimum measurement uncertainty occurs when the individual contributions of these uncertainties are equal. At the optimum admittance level the resulting calibration uncertainty is $\pm 0.013M$ percent, where M is the calibration frequency in megahertz.

For measurements of unknowns having small Q-values there is an additional capacitance uncertainty of $\pm 0.06M/Qx$ percent, where Qx is the Q-value of the unknown. This uncertainty results from a 100-ps time constant uncertainty in the calibration of the phase angle of the bridge's conductance standard.

The major sources of uncertainty in the measurement of conductance using the small-admittance configurations are:

- ±0.02 percent and ±0.002-μS uncertainties in the calibration of the conductance standard
- ±20 microradians and ±5-ps time constant uncertainties in the calibration of the loss angle of the capacitance standard. Actually, part of the 5-ps uncertainty may be due to uncertainty in the time constant of the bridge ratio. These uncertainties are the major uncertainties in measuring large Q-values.
- $\pm 0.006M$-μS conductance resolution (corresponding to ±0.001 pF), where M is the frequency in megahertz.
- $\pm 0.001\sqrt{M}$-ohm uncertainty in the resistance of Z_{CH}.

5.3 Sources of uncertainty in small-impedance configuration measurements

For measurements using the small-impedance configurations, the uncertainties in the calibrations of the capacitance and conductance standards produce corresponding uncertainties in the measured admittance difference between the unknown and reference balances and in the calibration values for the series and shunt admittances, Y_K and Y_{ET} in Fig. 10. In addition, the measurement of the admittance difference between the balances has an end term uncertainty of ±2 parts per million of the series admittance, Y_K.

The end terms for the impedance difference between an unknown and a short circuit at the end of 0.1-μH connecting leads are ±[(0.2 to 0.7) + $0.016/M$]nH and ±[(0.2 to 0.5) + $0.6M$]mΩ. The minimum values pertain to components having impedances small compared to the impedance of the connecting leads. The maximum values are for component impedances large compared to that of the connecting leads.

VI. AUTOMATIC MAINTENANCE AIDS

The overall system includes automatic aids for maintaining the calibration of the bridge standards and the operation of the hardware. Of particular importance are the aids for calibration maintenance. Errors in the calibration typically do not interfere with the measurement process to cause error messages, as hardware faults do, and so the only way to have continuing confidence in COZY's accuracy is to check the calibration on a routine basis. However, calibration checks involve so many bridge balances and computations that to make them practical on a routine basis requires automation.

The calibration maintenance aids include a group of programs that check for changes in the admittances of the steps of the capacitance and conductance standards. One program does a step-up calibration of the bridge standards for up to six successive times. The average values and the scatters for the capacitance and conductance of each step are printed out. Examination of these results shows whether the system is performing well and whether the reproducibilities of the steps are satisfactory.

Another program performs an element by element subtraction between the values of a new calibration and of the calibration being used for measurements. Examination of the results shows whether the calibration being used is satisfactory.

Still another program automatically measures the series and shunt admittances, Y_K and Y_{ET}, of the small-impedance configurations. These results can be compared with the existing calibrations.

The aids for maintaining the hardware include monitors at the 13 power supplies and at 29 points within the generator and receiver. The

outputs of the monitors can be read by the A/D converter and are used by a hardware checking program to measure: the voltages of the power supplies; the gains and losses of the amplifiers and attenuators; the in-band transmissions and out-of-band rejections of the low pass filters; and various signal levels. The program can be run to check that the functions and levels are within limits or to print out the results of the measurements. A go/no-go check of the functions and levels takes five minutes. When the generator or receiver is malfunctioning, a diagnosis of the measurements can lead to localizing the fault to within four active components.

A second hardware checking program tests the operation of the relays in the small-step decades of the bridge's and step-up unit's capacitance and conductance standards. The special mercury-wetted switches, which use platinum leads, are not as reliable as the standard switches, which use only magnetic alloy leads; the platinum-to-glass seal is fragile. As a result, some relays have failed. Typically these relays were in the bridge's small-step decades, which contain the most used relays. A few of these failed relays gave troubles by becoming slower than the settling times used in the measuring program. In these cases, static tests of relay operation were not sufficient.

VII. MEASUREMENTS OF ENVIRONMENTAL COEFFICIENTS AND DISACCOMMODATION FACTORS

7.1 Environmental coefficient measurements

To make environmental coefficient measurements the user connects the components to sample boards in the environmental chamber, connects a coaxial cable from the sample boards to the bridge's binding posts and enters via dialogue the desired measurement conditions, post processing of the measurement results and output media. Up to six components may be connected to each of three sample boards, with all the components on a board having the same nominal impedance. One signal level and up to 20 test frequencies may be specified for each board. Up to 20 environmental conditions, i.e., combinations of dry bulb temperature and relative humidity, may be specified. For each environmental condition there may also be specified a soak time, and whether the rate of temperature change is to be kept below 1° Celsius per minute (to avoid thermal shock).

For each environmental condition the computer passes to the chamber's microprocessor the dry bulb temperature, dew point, soak time, and whether the rate of temperature change is to be restricted. Then the computer repetitively queries the microcomputer concerning whether the specified conditions have been met. When the conditions have been met, COZY makes the specified impedance/admittance measurements and the computer then passes to the microcomputer the

next set of conditions. While the environmental conditions are being achieved, the computer and bridge unit are free for general purpose measurements.

Each of the three sample boards in the environmental chamber has eight pairs of binding posts. Six of the eight pairs are for samples, the seventh pair has a short circuit across it and the eighth pair is left open. The short-circuited and open-circuited binding posts are used to obtain corrections for the effects of the series impedances and shunt admittances of the cables, switches and sample board circuitry between the bridge's binding posts and the binding posts to which the samples are connected.

The sample boards are made from teflon impregnated sheets of woven fiberglass. Within the chamber the upper side of each quarter-inch thick board is plated with 3-mil copper that serves as the ground plane for eight strip lines on the board's under side. The strip lines provide the connections to the board's "high" binding posts. The thermal conductivity from the binding posts to the outside of the chamber is low enough so that at steady state the temperatures of the binding posts are within 0.1° Celsius of the chamber's temperature. The difference between the inductance and resistance of the strip line to a board's short-circuited binding posts and the inductances and resistances of the strip lines to the board's sample positions are corrected for in data reduction via stored values for these differences. Similarly, the capacitance differences between the open-circuited strip line and the strip lines to the sample positions are corrected for in data reduction. However, the capacitance differences vary nonreproducibly with temperature, humidity, and recent history. These nonreproducible variations cannot be corrected for and therefore introduce errors in environmental coefficient measurements. The worst case variation over the environment range of the chamber is ±0.2 pF.

Increased speed for environmental coefficient measurements was achieved by changing the software to take advantage of the similarities of many of the measurements. All the samples on one board are required to be nominally the same. Also, the impedances of the samples do not change drastically with the changes in environmental conditions. Consequently, the complete measuring process is used only for the first environmental condition and then only for the short-circuit, the open-circuit, and the first sample positions on each sample board. For the second through sixth sample positions only the unknown balances are made; the bridge configuration information and reference balance data are retained from the first sample position. For the second and following environmental conditions the bridge configuration information obtained during the first environmental condition is used to set up the bridge for balancing. As a result of the decreased

number of reference balances and determinations of approximate values for the unknowns, the average time for a measurement is less than 10 seconds, versus about 20 seconds for a general purpose measurement.

Increased accuracy for environmental coefficient measurements was achieved by changing the software so that measurements of small changes in admittance will not involve changes in the settings of decades having relatively large steps. Otherwise, small fractional changes in the admittances of the large-step decades could cause large errors in the measurements of small changes in admittance. This avoidance of changes in the large-step decades is done by making two balances instead of one when the bridge can be balanced at more than one setting of the standards. In these cases one balance uses zeros in the setting and the other balance uses tens. Thus, if a balance could be made at a setting of 2-0-0.-0-0 (i.e., the 100-pF per step decade set to two; the 10-pF, to zero; the 1-pF, to zero; etc.), it would be. However, a second balance would also be made at a setting of 1-9-9.-9-10. (For simplicity, in this example it is assumed that the actual value of a setting equals its nominal value.) The setting of the standards and the admittance at balance would be saved for both balances. Then, if at the next temperature the setting at balance were higher, say 2-0-0.-0-9, the difference would be calculated with respect to the 2-0-0.-0-0 setting. But if the setting were lower, say 1-9-9.-9-1, the difference would be computed with respect to the 1-9-9.-9-10 setting.

7.2 Disaccommodation factor measurements

Measurements of the disaccommodation factors of ferromagnetic materials are also made using the environmental chamber. The user specifies temperature, relative humidity, soak time, temperature rate restriction, measurement frequency, and whether measurements are wanted 100 and 1000 minutes after demagnetization. (Measurements 1 and 10 minutes after demagnetization are always made.) In addition, for each sample board the user specifies peak demagnetization current and test signal level.

In the measurements the temperature is held fixed at the user-specified value, the components are individually demagnetized, and their inductances are measured at the appropriate times after demagnetization. Demagnetization is done with a 60-Hz current that decreases linearly to zero in 15 seconds. The software for these measurements makes use of the zero- and ten-settings to optimize the accuracy for small admittance changes. Also, the software includes a reorganization of the major blocks of the measurement process to make the time sequencing of demagnetizations and measurements so efficient that the demagnetizations and one-minute measurements of all 18

samples are done before the first ten-minute measurement is to be made.

VIII. SUMMARY

The computer-operated impedance/admittance bridge (COZY) is an easy-to-use facility that provides fast, highly accurate measurements over a wide impedance range at frequencies between 200 Hz and 30 MHz. The accuracy is comparable to that achieved with specially developed manual bridges, but the expertise, care, and time required of the user are a tenth to a hundredth of that required for manual bridges. Also, the expertise and effort required to check and maintain the calibrations are similarly less. This is especially true since COZY's admittance-frequency coverage exceeds that of five manual bridges.

In addition to high accuracy, COZY also provides high resolution for measuring small changes in impedance/admittance. This attribute has proved to be very valuable in studies of stability versus shock and vibration and also versus time, and has been capitalized on by the addition of hardware and software that provide automatic measurements of environmental coefficients and disaccommodation factors.

The speed, accuracy, and resolution of COZY have been extensively used in the development and evaluation of materials, structures, and complete components for ferromagnetic inductors and transformers. In this work COZY has permitted measurements and evaluations that would have been otherwise impossible.

IX. ACKNOWLEDGMENTS

The computer-operated impedance/admittance bridge was developed by the Measuring Systems Design Department at Bell Laboratories and benefited greatly from the overall experience within the Department. Especially important were the contributions by O. Kummer, who developed the generator and receiver, and C. A. Reinhardt, who developed the hardware and firmware for the environmental coefficient and disaccommodation factor measurements. Significant contributions were also made by J. H. Churchill, F. R. Martinez, J. M. Neil, D. G. Neumann, R. L. Nichols, R. A. Noel, M. F. Pucci, and K. D. Sausser. The special mercury-wetted switches were developed and produced at Western Electric-Kansas city under the direction of J. P. Prior, C. K. Bell, and J. R. Bennett.

REFERENCES

1. L. D. White, "Accurate Immittance Measurements at Frequencies up to 20 MHz," IEEE Trans. Instru. Meas., *IM-19*, No. 4 (November 1970), pp. 331–6.
2. W. J. Geldart, G. D. Haynie, and R. G. Schleich, "A 50 Hz to 250 MHz Computer

Operated Transmission Measuring Set," B.S.T.J., *48,* No. 5 (May—June 1969), pp. 1339–81.
3. L. E. Huntley, "Some Applications of Series Impedance Elements in Radio Frequency Immittance Measurements," J. Res. N.B.S., *74C,* Nos. 3 and 4 (July–December 1970), pp. 79–85.
4. L. D. White and H. T. Wilhelm, "A 0.1 to 10 MHz Dielectric Specimen Bridge with Dissipation Factor Accuracy of $\pm 10^{-6}$," IEEE Trans. Instru. Meas., *IM-15,* No. 4 (December 1966), pp. 293–8.
5. L. H. Ford and N. F. Astbury, "A Note on the Calibration of Decade Condensers," J. Sci. Instru., *15* (1938), pp 122–6.
6. Thomas L. Zapf, "Capacitor Calibration by Step-Up Methods," J. Res. N.B.S., *64C,* No. 1 (January–March 1960), pp. 75–9.
7. W. D. Voelker, "An Improved Capacitance Bridge for Precision Measurements," Bell Laboratories Record, *20* (January 1942), pp. 133–7.

Contributors to This Issue

R. W. Coons, Western Electric Company, 1942–1948; Bell Laboratories, 1948—. Mr. Coons has worked mainly on the design and calibration of precision impedance measuring equipment. He was also involved in the design of microwave adjustable attenuators for the TH frequency band. Recently he has been working on a bridge applique to COZY that will provide improved accuracy from 2 to 30 MHz.

Narain Gehani, B. Tech., 1969, Indian Institute of Technology; M.S., 1975, Ph.D. (computer science), 1975, Cornell University; State University of New York at Buffalo, 1975–1978; Bell Laboratories, 1978—. From September 1975 to June 1978, Mr. Gehani was an Assistant Professor at State University of New York at Buffalo. His research and consulting interests include programming methodology and language design, office automation, concurrent programming, program correctness, compilers, data structures, and specification techniques.

O. Johnsen, Diploma in E.E., 1974, Ph.D., 1979, Ecole Polytechnique Federale de Lausanne, Switzerland; Ecole Polytechnique Federale de Lausanne, 1974–1979; Bell Laboratories, 1979—. Mr. Johnsen was assistant at the Signal Processing Laboratory of the Ecole Polytechnique Federale de Lausanne, working in the field of picture coding. At Bell Laboratories, as a member of the Visual Communications Research Department, he is involved in research in picture processing and coding and in communication systems.

Cory Myers, B.S., M.S. (electrical engineering and computer science), 1980, Massachusetts Institute of Technology, Cambridge; Bell Laboratories, 1977—. At Bell Laboratories, Mr. Myers initially worked on computer graphics, digital circuit design, and dynamic programming for speech recognition. He is currently in the digital signal processing group, where his interests include speech processing, recognition, and digital signal processing.

Kurt Nassau, B.Sc. (physics and chemistry), 1948, University of Bristol, England; Ph.D. (chemistry), 1959, University of Pittsburgh; Glyco Products Co., Inc., Williamsport, Pa., 1948–1954; Walter Reed Army Medical Center, Washington, DC, 1954–1956; Bell Laboratories, 1959—. He has worked in the areas of crystal chemistry, lasers, ferroelectric and related crystals, crystal growth and the clarification of the role of convection in Czochralski pulling, the origin of color in crystals including irradiation-induced colors, and the preparation of novel glasses by rapid quenching. Currently he is investigating glasses for long-distance optical waveguides. Member, American Chemical Society, American Crystallographic Association, American Association for Crystal Growth, Phi Lambda Upsilon; Fellow, Mineralogical Society of America.

Arun N. Netravali, B. Tech. (Honors), 1967, Indian Institute of Technology, Bombay, India; M.S., 1969, Ph.D. (electrical engineering), 1970, Rice University; Optimal Data Corporation, 1970–1972; Bell Laboratories, 1972—. Mr. Netravali has worked on problems related to filtering, guidance, and control for the space shuttle. At Bell Laboratories, he has worked on various aspects of signal processing. He is presently Head of the Visual Communication Research Department and a Visiting Professor in the Department of Electrical Engineering at Rutgers University. Member, Tau Beta Pi, Sigma Xi; Senior Member, IEEE.

Irwin W. Sandberg, B.E.E., 1955, M.E.E., 1956, and D.E.E., 1958, Polytechnic Institute of Brooklyn; Bell Laboratories, 1958—. Mr. Sandberg has been concerned with analysis of radar systems for military defense, synthesis and analysis of active and time-varying networks, several fundamental studies of properties of nonlinear systems, and with some problems in communication theory and numerical analysis. His more recent interests include macroeconomics, compartmental models, the theory of digital filtering, and global implicit-function theorems. Former Vice Chairman IEEE Group on Circuit Theory, and Former Guest Editor IEEE Transactions on Circuit Theory Special Issue on Active and Digital Networks. Fellow and member, IEEE; member, American Association for the Advancement of Science, Eta Kappa Nu, Sigma Xi, and Tau Beta Pi.

Robert C. Strum, B.S. (engineering science), 1966, Pennsylvania State University; M.Eng. (electrical engineering), 1967, Cornell University; M.S. (systems engineering), 1979, University of Pennsylvania;

Bell Laboratories, 1966—. Mr. Strum has worked in the areas of impedance calibration and computer-controlled measurements of impedance. He has also been involved with manual and computer-controlled measurements in fiber optics. His current activity concerns computer-based control systems for digital satellite networks. Member, IEEE, Tau Beta Pi.

Lowell D. White, A.B. (physics), 1949, M.A. (physics), 1951, Ph.D. (physics), 1956, Princeton University; Princeton University, 1953–1955; Bell Laboratories, 1955—. Mr. White's early work included research in traveling wave tubes and ammonia masers, followed by development of millimeter-wave transmission measuring sets. Since 1966, he has been involved primarily in the development of impedance standards and measuring equipment for use in developing and characterizing components, particularly ferromagnetic inductors, for transmission systems. Currently, Mr. White supervises a group concerned with test facilities for a lightwave undersea cable system. Member, American Physical Society, IEEE.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 3, March 1981
Printed in U.S.A.

Papers by Bell Laboratories Authors

PHYSICAL SCIENCE

Atmospheric Photochemistry. T. E. Graedel, in *The Handbook of Environmental Chemistry*, edited by O. Hutzinger, Heidelberg: Springer-Verlag, 1980, pp 107–43.

Brownian Dynamics Study of Polymer Conformational Transitions. E. Helfand, Z. R. Wasserman, and T. A. Weber, Macromolecules, *13* (May–June 1980), pp 526–33.

Crystallization and Morphology of Melt-Solidified Poly(vinylidene Fluoride). A. J. Lovinger, J Polym Sci Polym Phys Ed, *18* (1980), pp 793–809.

The Effect of Oxygen, Salt Concentration, and pH on Galvanic Cells Between Carbon Filled Polyethylene Cathode and Various Metal Anodes. G. Schick, ASTM J Test Eval, *8*, No. 3 (May 1980), pp 143–54.

Electrochemical Catalysis by Foreign Metal Adatoms. J. D. E. McIntyre and W. F. Peck, Jr., Proc 3rd Symp Electrode Proc, Electrochem Soc, PV80-3 (1980), pp 322–49.

Field-Induced Atomic Displacements in Crystals. S. C. Abrahams, Annu Yugoslav Centre Crystallography, *14* (1979), pp 1–12.

The Growth and Characterization of Very Thin Silicon Dioxide Films. A. C. Adams, T. E. Smith, and C. C. Chang, J Electrochem Soc, *127* (August 1980), pp 1787–94.

Iterative Calculation of Reverberation Time. M. R. Schroeder and D. Hackman, Acustica, *45*, No. 4 (1980), pp 269–73.

Optically Generated Pseudo-Stark Effect in Ruby. P. F. Liao, A. M. Glass, and L. M. Humphrey, Phys Rev B, *22* (September 1980), pp 2276–81.

Orientation Dependence of Breakdown Voltage in GaAs. M. H. Lee and S. M. Sze, Solid State Electron, *23* (September 1980), pp 1007–9.

Photoemission from the f^{12} and f^{13} Configurations. G. K. Wertheim, Chem Phys Lett, *72* (15 June 1980), pp 518–21.

Picosecond Optoelectronic Detection, Sampling, and Correlation Measurements in Amorphous Semiconductors. D. H. Auston, A. M. Johnson, P. R. Smith, and J. C. Bean, Appl. Phys. Lett., *37*, No. 4 (15 August 1980), pp 371–3.

The Regiospecific Formation of ^{13}C-Labelled Benzoins. S. H. Bertz, J Chem Soc Chem Commun (1980), pp 1831–2.

Relation of Drift Velocity to Low-Field Mobility and High-Field Saturation Velocity. K. K. Thornber, J Appl Phys, *51* (April 1980), pp 2127–36.

Role of Point Defects in the Growth of the Oxidation Induced Stacking Faults (OISF) in Silicon. II. Retrogrowth, Effects of Hydrogen Chloride Oxidation, and Orientation. S. P. Murarka, Phys Rev B, *21*, No. 2 (1980), pp 692–701.

Spectroscopy of Collective Pair Excitations. P. A. Fleury, in G. K. Horton and A. A. Maradudin, Eds., *Dynamical Properties of Solids*, Vol. III, New York: North Holland, 1980, pp 197–244.

Tensile Properties and Morphology of Blends of Polyethylene and Polypropylene. A. J. Lovinger and M. L. Williams, J Appl Poly Sci, *25* (1980), pp 1703–13.

Theory of Saturation Spectroscopy Including Collisional Effects. P. R. Berman, P. F. Liao, and J. E. Bjorkholm, Phys Rev A, *20* (December 1979), pp 2389–404.

Time Resolved Molecular Electronic Energy Transfer into a Silver Surface. R. Rossetti and L. E. Brus, J Chem Phys, *73* (1980), pp 572–7.

MATHEMATICS

On Constant Weight Codes and Harmonious Graphs. R. L. Graham and N. J. A. Sloane, *Proceedings West Coast Conference on Combinatorics, Graph Theory, and Computing*, Winnipeg: Utilitas Matematica, 1980, pp 25–40.

Data Analytic Displays for Ridge Regression. R. L. Obenchain, Tech Rep 605, Univ of Wisconsin, Dept of Statistics (April 1980).

Ergodic and Recurrence Properties of Generalized Semi-Markov Processes (abstract only). L. D. Fossett, Proc ORSA/TIMS Joint Natl Meeting, *10* (September 1980), p 76.

Informational Aspects of Stochastic Control. H. S. Witsenhausen, in Analysis and Optimization of Stochastic Systems, edited by Jabobs et al., New York: Academic, 1980, pp 272–84.

On Intersections of Interval Graphs. H. S. Witsenhausen, Discrete Math, *31* (August 1980), pp 211–6.

Problem E2741. H. S. Witsenhausen, Am Math Mon, *87* (January 1980) (solution). [Refer to Am Math Mon, *85* (November 1978), p 765 (problem).]

Representation and Approximation of Noncooperative Sequential Games. W. Whitt, SIAM J Control Optim, *18* (January 1980), pp 33–48.

Some Useful Functions for Functional Limit Theorems. W. Whitt, Math Op Res, *5* (February 1980), pp 67–85.

COMPUTING

Planning for the Bell Operations Systems Network. J. J. Amoss, Proc 5th Int Conf Computer Commun (October 80), pp 559–63.

What is an Online Search? D. T. Hawkins and C. P. Brown, Online, *4*, No. 1 (January 1980), pp 12–8.

ENGINEERING

Alpha-Particle-Induced Soft Errors and 64K Dynamic RAM Design Interaction. R. J. McPartland, J. T. Nelson, and W. R. Huber, Proc Int Reliability Physics Symp, 1980 (April 8–10, 1980), pp 261–7.

Ambipolar Transport in Double Heterostructure Injection Lasers. P. J. Anthony and N. E. Schumaker, IEEE Electron Device Lett, *EDL-1*, No. 4 (April 1980), pp 58–60.

Analysis of Microsegregation in Crystals. L. O. Wilson, J Cryst Growth, *48* (March 1980), pp 363–6.

Constant-Amplitude Antenna Arrays with Beam Patterns Whose Lobes Have Equal Magnitudes. M. R. Schroeder, Arch Elektr Ubertragung, *34* (1980), pp 165–8.

Digital Transmission of Commentary-Grade (7 KHz) Audio at 56 or 64 kb/s. J. D. Johnston and D. J. Goodman, IEEE Trans Commun, *COM-28*, No. 1 (January 1980), pp 136–8.

Dissolution and Formation of Intermetallics in the Soldering Process. W. G. Bader, Proc Seminar Phys Metallurgy Metals Joining (1980), pp 257–68.

Effect of Sidewalls on Wavenumber Selection in Rayleigh-Binard Convection. M. C. Cross, P. G. Daniels, P. C. Hohenberg, and E. D. Siggia, Phys Rev Lett, *45* (15 September 1980), pp 898–901.

Electroless Gold Plating on III-V Compound Crystals. L. A. D'Asaro, S. Nakahara, and Y. Okinaka, J Electrochem Soc, *127*, No. 9 (September 1980), pp 1935–40.

Fiber Measurement Standards. A. H. Cherin and W. B. Gardner, Laser Focus, *16*, No. 8 (August 1980), pp 60–5.

The Free Volume Concept and Its Implications on Dilatation in Glassy Polymers Under Shear Stresses. T. T. Wang and S. Matsuoka, J Polym Sci Polym Lett Ed, *18* (September 1980), pp 593–8.

FT3-The Bell System's Metropolitan Trunk Lightwave System. I. Jacobs and J. R. Stauffer, Sixth European Conference on Optical Communication (Pub. No. 190), New York: Institution of Electrical Engineers (1980), pp 431–4.

Geometric Effects on the Gate-Controlled Capacitor. E. B. Slutsky and J. N. Zemel, IEEE Trans Electron Devices, *ED-27* (September 1980), pp 1843–6.

Improving the Properties of Recycled Plastics—Successful Case Histories. G. H. Bebbington, Soc Plastics Engineers—Regional Tech Conf (SPE-RETEC) Preprints #5 (October 7, 1980).

Increasing the Wettability of Polymeric Battery Separators to Nonaqueous Electrolytic Solutions by Cross-Linking with Active Species in a Glow Discharge. J. J. Auborn and H. Schonhorn, Proc Symp Lithium Batteries, Electrochem Soc (October 6–10, 1980).

Inhomogeneous Thermal Degradation of Poly(vinylidene Fluoride) Crystallized From the Melt. A. J. Lovinger and D. J. Freed, Macromolecules, *13* (1980), pp 989–94.

Laboratory Testing of New Feature Generics. R. C. Eisele, Proceedings of the National Electronics Conference 1980, *34* (1 October 1980), pp 294–7.

400Å Linewidth Lithography on Thick Silicon Substrates. R. E. Howard, E. L. Hu, L. D. Jackel, P. Grabbe, and D. M. Tennant, Appl Phys Lett, *36* (April 1980), pp 592–4.

Low Cobalt CrCoFe and CrCoFe-X Permanent Magnet Alloys. M. L. Green, R. C. Sherwood, G. Y. Chin, J. H. Wernick, and J. Bernardini, IEEE Trans Mag, *MAG-16*, No. 5 (September 1980), pp 1053–5.

A Low-High Junction Solar-Cell Model Developed for Use in Tandem Cell Analysis. R. I. McPartland and A. G. Sabris, Solid State Electronics, *23* (June 1980), pp 605–10.

Low Leakage Current and Saturated Reverse Characteristic in Broad-Area Indium and Gallium and Arsenide and Phosphide Diodes. F. Capasso, R. A. Logan, P. W. Foy, and S. Sumski, Electron Lett, *16*, No. 7 (March 1980), pp 241–2.

Miniature Packaged Crystal Oscillators. R. E. Paradysz, D. M. Embree, U. R. Saari, and R. J. McClure, Proceedings of the 34th Annual Symposium on Frequency Control 1980 (1980), pp 475–87.

A New Family of Ferroelectric Materials With Composition $A_2BMO_3F_3$. J. Ravez, G. Peravdeau, H. Arend, S. C. Abrahams, and P. Hagenmuller, Ferroelectrics, *26* (March 1980), pp 767–9.

An NMOS Analog Building Block for Telecommunications Applications. P. E. Fleischer, K. R. Laker, D. G. Marsh, J. P. Ballantyne, and A. A. Yiannoulos, IEEE Trans Circuits Systems, *CAS-27* (June 1980), pp 552–9.

A Note on the Effect of Incident Beam Convergence on Quantitative Electron Energy Loss Spectroscopy. D. C. Joy, D. M. Maher, and R. C. Farrow, *Microbeam Analysis 1980*, edited by D. B. Wittny, San Francisco, California: San Francisco Press, 1980, pp 154–7.

Nuclear Magnetic Resonance Analysis of the Microstructure of Poly(chloroprene sulfone). R. E. Cais and G. J. Stuk, Macromolecules, *13* (1980), pp 415–26.

Observations of Negative Resistance Associated With Superlinear Emission Characteristics of (Al,Ga)As Double-Heterostructure Lasers. P. J. Anthony, T. L. Paoli, and R. L. Hartman, IEEE J Quantum Electron, *QE-16*, No. 7 (July 1980), pp 735–9.

An Optimal Design of LSI CMOS Polycells. S. M. Kang, Proc 1980 IEEE Int Symp Circuits Systems, *3* (April 1980), pp 1008–10.

Optimization of Concatenated Fiber Bandwidth Via Differential Mode Delay. M. J. Buckler, Tech Dig, Symp Optical Fiber Measurements, 1980 (October 190), pp 59–62.

Optimization of Optoacoustic Cell for Depth Profiling Studies of Semiconductor Surfaces. A. C. Tam and Y. H. Wong, Appl Phys Lett, *36*, No. 5 (15 March 1980), pp 471–3.

PBS—Positive Electron Resist—Capabilities and Limitations. M. J. Bowden, R. F. W. Pease, L. D. Yau, J. Frackoviak, L. E. Thompson, J. G. Skinner, and J. P. Ballantyne, in H. Ahmed and W. C. Nixon, Eds., *Microcircuit Engineering*, Cambridge: Cambridge U.P., pp 239–54.

Photon Correlation Spectroscopy Near the Glass Transition in Polymers. G. D. Patterson and J. R. Stevens, ACS Polym Preprints, *21*, No. 2 (1980), pp 16–7.

Photon Correlation Spectroscopy of Polystyrene Solutions. G. D. Patterson, J.-P. Jarry, and C. P. Lindsey, Macromolecules, *13* (1980), pp 668–70.

Physical Level Protocols. H. V. Bertine, IEEE Trans Commun, *COM-28*, No. 4 (April 1980), pp 433–44.

Pumping Hazardous Gases. D. B. Fraser, J. L. Vossen, D. M. Hoffman, H. L. Pinch, R. M. Brown, M. Baron, and L. F. Dahlstedt, Am Vac Soc Mono, Pumping Hazardous Gases (June 1980).

Rapid Interferometric Examination of Glass for Index Inhomogeneity. A. D. White, Appl Optics, *18* (1 August 1979), p 2525.

Real Time Simulation of a Multichannel Interframe Coder for Video Conferencing. A. B. Larsen, E. F. Brown, and J. M. Santelle, 1980 Canadian Commun & Power Conf (October 1980), pp 485–8.

Signal and Noise Response of a Spectrum Expander. H. E. Rowe, IEEE Trans Circuits Systems, *CAS-27* (September 1980), pp 804–15.

Science and Technology in Lightwave Telecommunications. S. E. Miller, Telecommun J, *47* (June 1980), pp 375–8.

Single Crystal AgBr Infrared Optical Fibers. T. J. Bridges, J. S. Hasiak, and A. R. Strnad, Optics Lett, *5* (March 1980), pp 85–6.

Solid Phase Solder Bonding for Use in the Assembly of Microelectronic Circuits. R. H. Minetti, Proc 1980 Int Microelectron Symp, pp 126–9.

Solution to a Solderability Problem by Combining Metallography and Thermal Analysis. J. E. Bennett, T. M. Paskowski, and H. E. Bair, in *Proceedings of the Tenth NATA Conference*, Boston, Massachusetts: North American Thermal Analysis Soc, 1980, pp 185–91.

Superconductivity and Resistivity of Amorphous Niobium Germanium Alloys. S. Kosinki, Phys Lett, *76A*, No. 2 (March 80), pp 160–7.

Techniques of Ultra Fine Pattern Generation. R. E. Howard, Solid State Technol, *23* (August 1980), pp 127–32.

Temperature Dependence of the Losing Threshold Current of Double Heterostructure. P. J. Anthony and N. E. Schumaker, J Appl Phys, *51*, No. 9 (September 1980), pp 5038–40.

Textured Thin-Film Si Solar Selective Absorbers Using Reactive Ion Etching. H. G. Craighead, R. E. Howard, and D. M. Tennant, Appl Phys Lett, *37* (1 October 1980), pp 653–5.

Thermal Shearing Effects on the Temperature Stability of Saw Devices. R. L. Rosenberg, IEEE Trans Sonics & Ultrasonics, *SU-27* (May 1980), pp 130–3.

Thermocompression Bondability of Thick Film Gold, A Comparison to Thin Film Gold. N. T. Panousis and R. C. Kershner, Proc 1980 Electron Components Conf (April 1980), pp 472–7.

On the Thickness and Spatial Distribution of a Fluoropolymer Film Deposited by Solution Dipping. H. G. Tompkins and S. P. Shorma, J Apply Polym Sci, *25* (February 1980), pp 211–222.

Transport Properties of sn-doped $Al_xGa_{1-x}As$ Grown by Molecular Beam Epitaxy. H. Morkog, A. Y. Cho, and C. Radice, Jr., J Appl Phys, *51*, No. 9 (September 1980), pp 4882–4.

Trap-Controlled Transient Photoconductivity in Dielectrics. R. J. Fleming, J Appl Phys, *50* (1979), pp 8075–81.

Two-Dimensional Interaction of Ion-Acoustic Solitons. P. A. Folkes, H. Ikezi, and R. Davis, Phys Rev Lett, *11* (15 September 1980), pp 902–4.

Ultrasmall Superconducting Josephson Junction. E. L. Hu, R. E. Howard, L. D. Jackel, L. A. Fetter, and D. M. Tennant, IEEE Trans Electron Devices, *ED-27* (October 1980), pp 2030–1.

Waveguide Propagation in Frozen Gas Matrices. R. Rossetti and L. E. Brus, Rev Sci Instrum, *51* (1980), pp 467–70.

SOCIAL AND LIFE SCIENCES

Acoustic Inverse Scattering as a Means for Determining the Area Function of a Lossy Vocal Tract: Theoretical and Experimental Model Studies. J. R. Resnick, J Acoust Soc Am, *67* (February 1980), p 722.

Acoustics in Human Co: munications: Room Acoustics, Music, and Speech. M. R. Schroeder, J Acoust Soc Am, *68*, No. 1 (July 1980), pp 22–8.

Application of Dynamic Time Warping to Connected Digit Recognition. L. R. Rabiner and C. E. Schmidt, IEEE Trans Acoustics, Speech, and Signal Processing, *ASSP-28* (August 1980), pp 377–88.

Application of Isolated Word Recognition to a Voice Controlled Repertory Dialer System. L. R. Rabiner, J. G. Wilpon, and A. E. Rosenberg, Proc 1980 Int Conf Acoustics, Speech, and Signal Processing (April 1980), pp 182–5.

Cochlear Macromechanics—Time Domain Solutions. J. B. Allen and M. M. Sondhi, J Acoust Soc Am, *66* (July 1979), pp 123–32.

Cochlear Models—1978. J. B. Allen, in *Workshop on Models of the Auditory System and Related Signal Processing Techniques*, Stockholm: Almquist and Wiksell, 1979, pp 1–16.

Computer Studies on Parametric Coding of Speech Spectra. J. L. Flanagan and S. W. Christensen, J Acoust Soc Am, *68*, No. 2 (August 1980), pp 420–30.

A Connected Digit Recognizer Based on Dynamic Time Warping and Isolated Digit Templates. L. R. Rabiner and C. E. Schmidt, Proc 1980 Int Conf Acoustics, Speech, and Signal Processing (April 1980), pp 194–8.

Considerations in Applying Clustering Techniques to Speaker Independent Word Recognition. L. R. Rabiner and J. G. Wilpon, Proc IEEE Int Conf Acoustics, Speech, and Signal Processing (April 1979), pp 578–81.

Experimental Studies in a New Automatic Speaker Verification System Using Telephone Speech. S. Furui and A. E. Rosenberg, Proc 1980 IEEE Int Conf Acoustics, Speech, and Signal Processing, *3* (April 1980), pp 1060–2.

Experiments with New Telecommunications Service Capabilities. R. E. Cardwell, ISSSL 80 Proc, IEEE Cat # 80 CH1565-1 (September 1980), pp 187–91.

A Model for Advanced Reservations for Large Scale Conferencing Services. H. Luss, J Op Res Soc, *31* (March 1980), pp 239–45.

On the Implementation of a Short-Time Spectral Analysis Method for System Identification. L. R. Rabiner and J. B. Allen, IEEE Trans Acoustics, Speech, and Signal Processing, *ASSP-28* (February 1980), pp 69–78.

Invertibility of a Room Impulse Response. S. T. Neely and J. B. Allen, J Acoust Soc Am, *66* (July 1979), pp 165–9.

An Overview of Speech Recognition Research at Bell Laboratories. S. E. Levinson, Second Annual IEEE Eng in Medicine and Biology Conf Proc (September 1980), pp 87–91.

Parametric Coding of Speech Spectra. J. L. Flanagan, J Acoust Soc Am, *68*, No. 2 (August 1980), pp 412–9.

Procedures for Conducting a Successful Telemeeting. C. Stockbridge and T. B. Bateman, Ninth Int Symp Human Factors in Telecommun (October 1980), pp 199–204.

Signal Models for Low Bit-Rate Coding of Speech. J. L. Flanagan, K. Ishizaka, and K. L. Shipley, J Acoust Soc Am, 68, No. 3 (September 1980), pp 780–91.

Speaker Independent, Isolated Word Recognition for a Moderate Size (54 Word) Vocabulary. L. R. Rabiner and J. G. Wilpon, IEEE Trans Acoustics, Speech, and Signal Processing, *ASSP-27* (December 1979), pp 583–7.

Speaker Independent Recognition of Isolated Words Using Clustering Techniques. L. R. Rabiner, S. E. Levinson, A. E. Rosenberg, and J. G. Wilpon, IEEE Trans Acoustics, Speech, and Signal Processing, *ASSP-27* (August 1979), pp 336–49.

Statistical Properties of the Log Likelihood Ratio for LPC. J. M. Tribolet, L. R. Rabiner, and M. M. Sondhi, IEEE Trans Acoustics, Speech, and Signal Processing, *ASSP-27* (October 1979), pp 550–555.

Technique for Frequency Division/Multiplication of Speech Signals. J. L. Flanagan and S. W. Christensen, J Acoust Soc Am, *68*, No. 4 (October 1980), pp 1061–8.

Techniques for Expanding the Capabilities of Practical Speech Recognizers. J. L. Flanagan, S. E. Levinson, L. R. Rabiner, and A. E. Rosenberg, in Wayne Lea, Ed., *Trends in Speech Recognition*, Englewood Cliffs, NJ: Prentice-Hall, 1980, pp 425–44.

MANAGEMENT AND ECONOMICS

An Extended Application of the Delta(2) Measure of Predictor-Variable Importance. P. E. Green, J. D. Carroll, and W. S. De Sarbo, Proc Am Market Assn Education Conf, 1979 (1979), pp 1–4.

SPC Network Planning in the Bell System. S. Horing, J. J. Lawser, and R. L. Simms, Jr., Proc Int Symp Telecommun Networks Plan (October 1980), pp 307–11.

Contents, April 1981

THE BELL SYSTEM TECHNICAL JOURNAL is abstracted or indexed by *Abstract Journal in Earthquake Engineering, Applied Mechanics Review, Applied Science & Technology Index, Chemical Abstracts, Computer Abstracts, Current Contents/Engineering, Technology & Applied Sciences, Current Index to Statistics, Current Papers in Electrical & Electronic Engineering, Current Papers on Computers & Control, Electronics & Communications Abstracts Journal, The Engineering Index, International Aerospace Abstracts, Journal of Current Laser Abstracts, Language and Language Behavior Abstracts, Mathematical Reviews, Science Abstracts (Series A, Physics Abstracts; Series B, Electrical and Electronic Abstracts; and Series C, Computer & Control Abstracts), Science Citation Index, Sociological Abstracts, Social Welfare, Social Planning and Social Development,* and *Solid State Abstracts Journal.* Reproductions of the Journal by years are available in microform from University Microfilms, 300 N. Zeeb Road, Ann Arbor, Michigan 48106.


Bell System