# RECONSTRUCTION OF MISSING PACKETS OF PCM AND ADPCM ENCODED SPEECH

by

Ondria G. Jaffe

SUBMITTED TO THE DEPARTMENT OF
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
OF THE DEGREES OF

BACHELOR OF SCIENCE

and

MASTER OF SCIENCE
IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 1986

Signature of Author____________________
Department of Electrical Engineering and Computer Science
June 2, 1986

Certified by____________________
Victor W. Zue
Thesis Supervisor
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology

Certified by____________________
David J. Goodman
VI-A Company Thesis Supervisor
AT&T Bell Laboratories

Accepted by____________________
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

# RECONSTRUCTION OF MISSING PACKETS OF PCM AND ADPCM ENCODED SPEECH

by

Ondria G. Jaffe

Submitted to the Department of Electrical Engineering and Computer Science
on June 2, 1986 in partial fulfillment of the requirements for the degrees of
Bachelor of Science
and
Master of Science in Electrical Engineering and Computer Science

## ABSTRACT

In a packetized voice communication network, if packets are dropped, late, or damaged because of congestion or interference, there can be severe degradation in speech quality at the receiver. If the speech was ADPCM coded, missing packets also cause the step-sizes, predictor coefficients, and prediction signals of the transmitter and receiver to diverge. My thesis explores several algorithms for reconstructing packetized PCM and ADPCM encoded speech at the receiver to improve quality. Results are verified by formal subjective testing.

The subjective tests show that for PCM speech, if a mean opinion score of at least three is desirable, the missing packet ratio should not exceed two percent, when the missing packets are replaced by silence. However, if the missing packets are reconstructed using one of the new algorithms, a missing packet ratio of eight percent can be tolerated. For ADPCM speech, it is tolerable to drop one percent of the packets when they are replaced with silence, whereas one of the new algorithms allows one to drop four percent.

Thesis Supervisor: Dr. Victor W. Zue

Title: Associate Professor of Electrical Engineering

VI-A Company Thesis Supervisor: Dr. David J. Goodman

Company: AT&T Bell Laboratories

## Acknowledgements

I appreciated the support and advice of David Goodman, Reinaldo Valenzuela, and Lawrence Wong in developing my algorithms. I thank Charles Dvorak, Howard Page, and Cecilia A. Ward for helping me to run the subjective tests. I also thank Duane O. Bowker for the source speech.

## CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

# 1. Introduction

## 1.1 Background

At present there are two types of communication networks: telephone networks that use circuit switching, and data networks that use packet switching. In circuit switching, a user is assigned a circuit for the duration of his call, and no other user can use that circuit during that time. In packet switching, a transmission is divided into packets, and each packet is sent through the network independently.

Lately, the push has been towards transmitting various types of information on common networks. Specifically, work is now being done on networks which allow both voice and data to be transmitted in packets.

In the case of packetized data, excessive network congestion or transmission errors result in delay. This is because it can take a packet longer to get through a congested network, or because a transmission error necessitated retransmission of the data. However, in the case of packetized speech, the same conditions cause missing packets. This is because speech must be played out at the receiver at a constant rate, or it will not sound correct. If a packet is late because of congestion, either something else must be played out during its time slot, or the speech must be speeded up by skipping the missing packet. Unlike data, a speech packet which is lost or damaged during transmission cannot be retransmitted, so that these problems also cause missing packets.

Unlike circuit switching networks, packet networks can take advantage of the burstiness of speech by implementation of speech activity detection (SAD). What is meant by burstiness is that a speech waveform is characterized by periods of high energy (speech bursts) separated by periods of low energy (silence). This occurs because of the nature of human conversation. When using SAD, silent packets are not transmitted at all, so that channel capacity can be increased by up to 100%. [1]

The biggest problem with transmitting packetized voice is the random delay that is introduced by packet queueing and network protocols. These delays complicate reconstruction at the receiver. When no packets are dropped in a network, reconstitution of speech is made possible by the timestamps and sequence numbers in the headers. [2] Both are needed in a network that implements SAD. Without the sequence numbers, if there were a large gap between two consecutive timestamps, it would be impossible to know at the receiver whether there is a missing packet between them, or simply silence. When SAD is used, silent packets are not transmitted, thus it is necessary for the receiver to be able to tell whether long gaps in the speech are due to missing speech, or actual silence. If one is substituting silence for missing packets, this is not a problem, because silence would do for both cases. However, if one is trying to reconstruct the missing packets, the sequence numbers are necessary. Conversely, without the timestamps, the sequence numbers indicate in what order to play out the packets, but not how much space to leave between them. Since a network implementing SAD does not transmit silent packets, the spacing between packets is not constant, and the receiver must see the timestamps in order to place them correctly.

Two important issues related to packetized speech transmission are what packet size to use, and what to do if a packet is lost in transmission or late at the receiver. The first of these requires consideration of packetization delay, perceptual effect of lost packets, network throughput and load, and packetization overhead. [2] For instance, in order to minimize the packtization delay, packets should be as short as possible. Going by this criterion, Rous and See suggested a packet size of 2-4 ms. [3]

The packets should also be of a size that will minimize the perceptual effect of lost packets. This criterion dictates a packet size smaller than 50 ms. [2] There has been some disagreement as to what an appropriate size is. Although Weinstein and Forgie feel that the shorter the packets, the less the perceptual effect when they are lost, Jayant and Christensen suggested that 32 ms packets are more robust to losses than the 4 ms packets mentioned above. Table 1 summarizes their description of the perceptual effects of lost packets of various sizes when they are replaced with silence: [4]

**TABLE 1.** perceptual effects of lost packets when they are replaced with silence

| Packet Length (ms) | Nature of Distortion |
|---|---|
| $\leq 4$ | crackles |
| 16-32 | glitches |
| $\geq 64$ | phoneme losses |

Based on by own listening experience, I would say that "crackles" are similar to the sound of crinkling up wax paper, while "glitches" describes a popping sound like blowing your bubble gum until it bursts. "Glitches" are perceptually better than "crackles" because for equal ratios of lost packets to total packets, the "glitches" from longer packets occur less frequently than the "crackles" of shorter packets.

In contrast to these two reasons to limit packet size, it is also important to make packets as long as possible relative to packetization overhead, in order to maximize channel utilization. Packetization overhead comes from headers containing timestamps, sequence numbers, source and destination addresses, and the like. [2] For packets which are the same length in time, narrowband speech has shorter packets in terms of bits than does wideband speech. Thus packet overhead is not as bad a problem for wideband speech.

One property of networks that constrains packet size is network throughput in packets/sec. [2] The shorter the packets, the higher the loads at network nodes, because more packets are used to send the same amount of speech. That is, there are more headers to sort through. This load can be increased until the upper limit of network throughput is reached. Better methods of reconstructing missing packets can make this limit less important, because one could drop packets at the transmitter and reconstruct them at the receiver, thus reducing load and effectively increasing throughput.

Obviously, there are many trade-offs between long and short packets. In real-time speech transmission experiments across ARPANET in 1983, Weinstein and Forgie used 100-200 ms packets, because of the constraints of network throughput. [2] Tucker and Flood, however, feel that 8 or 16 ms packets, a value very close to Jayant's and Christensen's preference, yield higher quality speech. [5]

The second issue is what to do if a packet is dropped, which could happen if it is lost in transmission or late at the receiver. When a packet is dropped, sustaining the flow of speech with a substitute packet or a silent packet is perceptually better than speeding up the speech in spots by playing out the next packet with no regard for the missing one. In general, the best choice of such a substitute varies with encoder type, packet size, and the statistics of gaps introduced by the network. [2]

Some work on this has been done in the past. In the ARPANET experiments, Weinstein and Forgie experimented with: (1) substituting silence for the missing packet; (2) repeating the previous packet; and (3) filling the gap with repeated frames of speech data, making the speech voiceless, and letting the energy values decay exponentially with time. They found that the third strategy was best when using a framed vocoder. Tucker and Flood preferred repeating the previous packet to substituting silence for the missing packet. [5] When the gaps are filled with silence, speech is substantially affected by loss ratios of 1 or 2 percent. [6] However, at the same loss ratios, using 16 or 32 ms packets, no significant effect can be observed from opinion scores when the gaps are filled by repeating the previous packet. [3]

Most of the above work was done with PCM signals. Jayant and Christensen have also done work with adaptive DPCM signals. [4] They used 32 ms packets in their experiments, and the quantizer step size was adapted once for each packet, and then held fixed for the duration of the packet. The step size was transmitted in the header, rather than computed at the receiver. When a packet was lost, they replaced it by silence, and then set the first sample of the following packet to zero. Table 2 shows their results.

**TABLE 2.** results of Jayant and Christensen's work in reconstructing missing packets of adaptive DPCM speech

| % Speech Loss | Perceptual Degradation |
|---|---|
| 0.2 | barely noticeable |
| 1.0 | noticeable to critical listener |
| 2.0 | noticeable |
| 5.0 | somewhat objectionable |
| 10.0 | definitely objectionable |

### 1.2 Thesis Overview

There is obviously a need for better ways to reconstruct missing packets of PCM and ADPCM encoded speech. In this thesis I present a new waveform substitution technique based on pattern matching, and I compare it to some previously tried techniques and a recently developed pitch detection technique in formal subjective tests.

In Section 2 I describe the reconstruction algorithms. Section 2.1 explores the algorithms for PCM speech, while Section 2.2 explores the algorithms for ADPCM speech. In section 3 I describe the formal subjective tests. Sections 3.1 and 3.2 respectively contain descriptions of the source speech and testing facilities. In sections 3.3 and 3.4 respectively, I explain how I designed the tests for the PCM algorithms and the ADPCM algorithms. Appendix A contains a copy of the instructions which were read to the subjects before they took the test. In Sections 4.1 and 4.2 I interpret the results of the two subjective tests. The detailed data analysis is in Appendix B. Finally, in the conclusion in Section 5, I restate the main results and suggest future research efforts.

## 2. Description of Algorithms

My approach to the reconstruction of lost packets is based on the fact that the speech signal does not change its characteristics too abruptly. It is short-term stationary, and voiced speech is also quasi-periodic. I observe that just before and just after each missing waveform segment, there are likely to be segments that are very similar to the missing one. Based on this observation I propose several closely related procedures for constructing new speech packets to replace the missing one.

### 2.1 Algorithms for PCM Speech

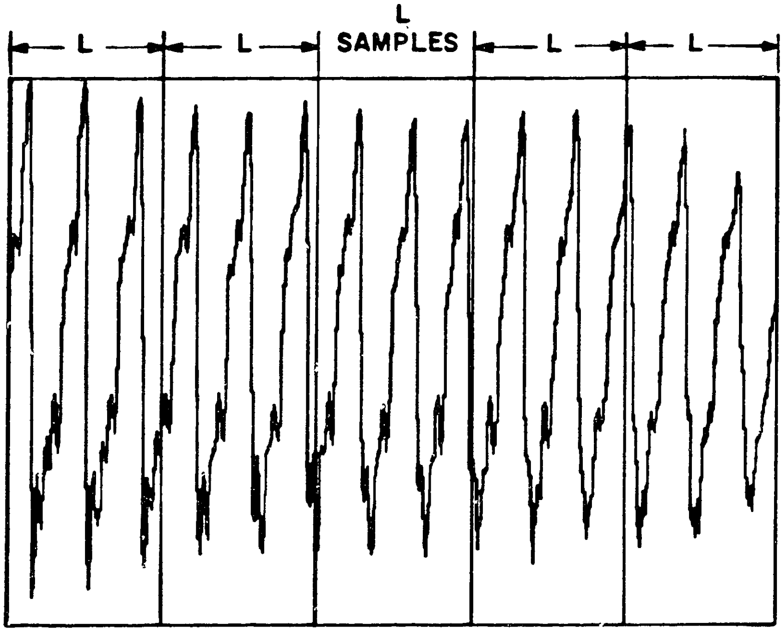
I present two methods for reconstructing PCM speech. The two-sided scheme uses speech that occurred both before and after the lost packet, while the one-sided scheme uses only speech from the past. The waveform segments to be used in the reconstruction are selected by pattern matching. I have investigated various pattern matching algorithms and found that the quality of the results is approximately constant across many of them. This provides flexibility in choosing a hardware implementation. I have simulated these reconstruction algorithms in the Fortran 77 programming language, under the *UNIX*$^{TM}$ operating system, on a VAX 11/780.

#### *2.1.1 Overview of the Two-sided Scheme*

For each missing packet, the algorithm scans two 'search windows' in order to identify speech segments resembling the missing packet. One search window precedes the missing packet and the other follows it. There are also two 'templates'. The templates frame two small blocks of speech, one immediately preceding the lost packet, and the other immediately following it. The algorithm compares the small block of speech in each template with the speech in the corresponding search window to find the best match. The template preceding the missing packet slides along the search window preceding the missing packet, and the algorithm identifies a segment of speech within the search window that best matches the template. The past reconstruction segment is the packet-length block of speech immediately following the match, plus two short segments of speech immediately preceding and following the block.

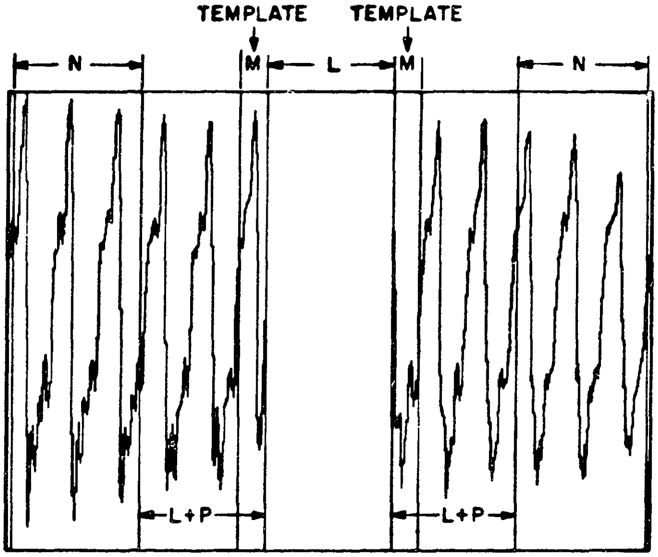
These short segments will merge with the received speech surrounding the missing packet. There is a similar search through future information, and the packet-length block of speech immediately preceding the best match, along with corresponding short merge segments, is chosen as the future reconstruction segment. The last steps are to combine the two reconstruction segments, normalize the energy of the combined packet estimate, and merge the ends of the estimate into the received speech surrounding the missing packet.

**Figure 1.** Speech waveform and packet boundaries. Each packet contains $L$ consecutive speech samples.



The parameters of this algorithm are the size of the packets ($L$ samples), the size of the templates ($M$ samples), the size of the search windows ($N$ samples), and the size of the merge windows ($P$ samples). These parameters and the algorithm itself are illustrated in Figures 1-5. Figure 1 shows a speech waveform divided into packets, each containing $L$ samples. Figure 2 shows the location of a missing packet and two templates. The past template frames the $M$ samples immediately preceding the lost packet, and the future template frames the $M$ samples immediately following the lost packet. In addition, notice in Figure 2 past and future search windows, each with $N$ samples. In order to leave room for a merge segment, there is a gap of $L+P$ samples between each search window and the missing packet. Figure 3 indicates that the speech in each template slides along the corresponding search window to determine the best match. In Figure 4, observe the $L$-sample segments to be used to construct a new packet, along with their $P$-sample merge segments. One of them follows the best match in the past search window, and the other precedes the best match in the future search window. Finally, Figure 5 shows the reconstructed packet obtained by combining the two segments in Figure 4.

**Figure 2.** One packet is missing and for the moment replaced by silence. There are two templates each containing $M$ samples, and two search windows with $N$ samples each. There is a gap of $L+P$ samples between each search window and the missing packet.



### *2.1.2 Pattern Matching*

There are several methods of pattern matching which yield almost equivalent results. One method is cross-correlation based on the sum of products of samples in the template and samples of the search window. This sum of products is divided by the energy of the $M$ samples in the search window segment. If the samples of the template are $x(i)$, and the samples of the search window are $y(i)$, then the cross-correlation formula is:

$$C(n) = \frac{\sum_{m=1}^{M} x(m)y(n+m)}{\sum_{m=1}^{M} [y(n+m)]^2} \tag{1}$$

where $M$ is the number of samples in the template and $n$ identifies the position of the template samples as they slide along the search window. Each time $C(n)$ is computed, the template moves along the search window by one more sample ($n$ is incremented by 1), and $C(n)$ is

**Figure 3.** Each template slides along the corresponding search window and the algorithm identifies a segment in the search window that best matches the samples in the template.

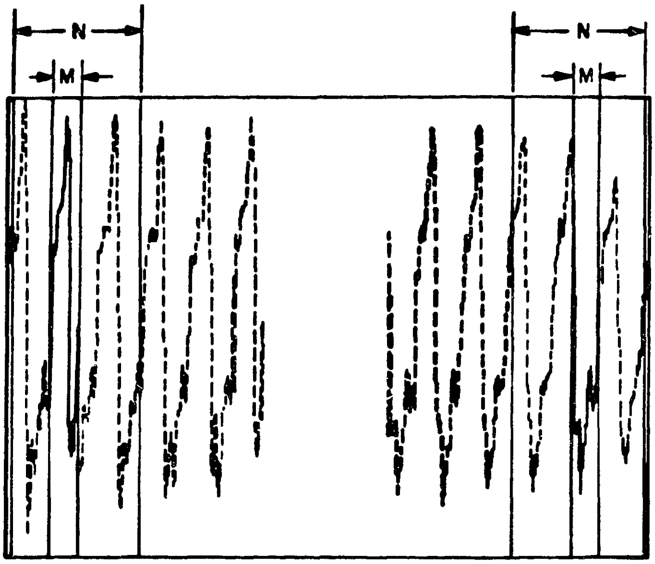
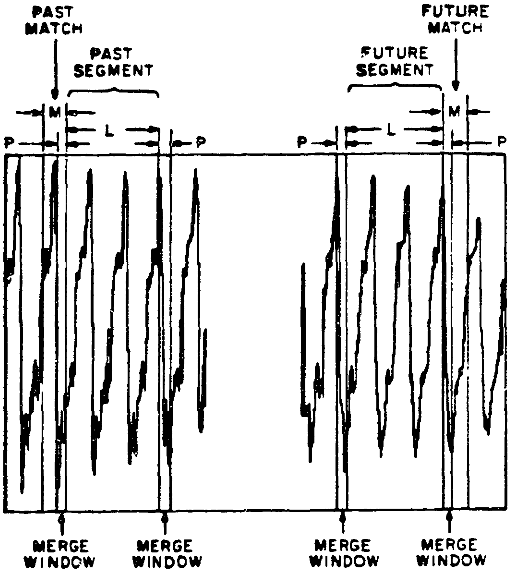


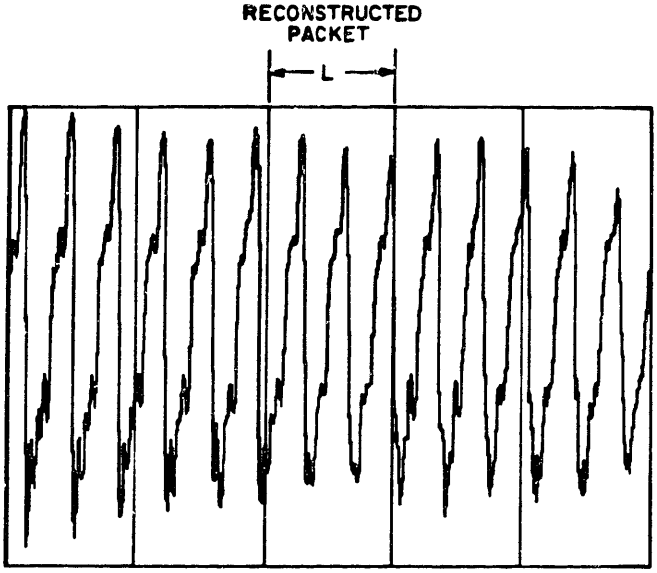
**Figure 4.** The $L$ samples following the best match to the past template comprise the past reconstruction segment. The $L$ samples preceding the best match to the future template comprise the future reconstruction segment. Each reconstruction segment is framed by merge segments.



computed again. The result of the search is the value of $n$ corresponding to the maximum $C(n)$.

A simplified version of $C(n)$, which is attractive in practical implementations, is the sign correlation,

**Figure 5.** The reconstructed packet is a combination of the future reconstruction segment and the past reconstruction segment.



$$S(n) = \sum_{m=1}^{M} sgn[x(m)]sgn[y(n+m)] \tag{2}$$

where $sgn(x)$ is $+1$ when $x>0$ and $-1$ when $x<0$.

Another approach to pattern matching is based on waveform differences. Following this approach I subtract the normalized samples in the template from normalized samples in the search window and sum up the absolute values of the differences. (Normalization algorithms will be discussed next.) As the template slides along the search window, the algorithm seeks the minimum sum of absolute differences.

In order that the result be sensitive to waveform shapes rather than level changes, the speech segments are normalized first. I have considered three methods of normalization. One is to divide the samples of each segment by the square root of the energy of that segment. This leads to the difference measure,

$$D_1(n) = \sum_{m=1}^{M} \left| \frac{x(m)}{\sqrt{\sum_{j=1}^{M}[x(j)]^2}} - \frac{y(n+m)}{\sqrt{\sum_{j=1}^{M}[y(n+j)]^2}} \right| \tag{3}$$

Note that the first denominator is independent of $n$ for a given missing packet, because the template never changes. However, the second denominator changes each time the template advances one sample. Another way to normalize is by the sum of the absolute magnitudes of the samples,

$$D_2(n) = \sum_{m=1}^{M} \left| \frac{x(m)}{\sum_{j=1}^{M} |x(j)|} - \frac{y(n+m)}{\sum_{j=1}^{M} |y(n+j)|} \right| \tag{4}$$

A third way is to divide by the peak-to-peak amplitude of the segment. In the case of the search window, the peak-to-peak amplitude is not across the whole window, but only across the segment being compared to the template. The formula for the resulting difference measure is:

$$D_3(n) = \sum_{m=1}^{M} \left| \frac{x(m)}{x_{\max} - x_{\min}} - \frac{y(n+m)}{y_{\max} - y_{\min}} \right| \tag{5}$$

where

$$x_{\max} = max[x(1), x(2), \ldots, x(M)],$$
$$y_{\max} = max[y(n+1), y(n+2), \ldots, y(n+M)]$$

and similarly for $x_{\min}$ and $y_{\min}$.

I conducted simulation experiments by listening to speech reconstructed using the various pattern-matching procedures. These experiments revealed that for the two-sided scheme, the difference methods produce speech that sounds slightly better than that produced with the correlation measures. The quality of the one-sided scheme (Section 2.1.5) seems to be insensitive to the pattern-matching measure. Within the difference methods, the method of normalization affects the choice of reconstruction packet, thereby modestly affecting the quality of the reconstructed speech. There is no perceptual difference between the square-root of the energy and the sum of absolute value normalizations, Equations (3) and (4) respectively, but the peak-to-peak normalization, Equation (5), produces noticeably worse results.

### *2.1.8 Combining Reconstruction Segments to Create a Reconstruction Packet*

In the two-sided scheme, the replacement packet is a combination of the two segments obtained from the pattern matching procedure. One way to combine the two normalized reconstruction segments is to average them. However, it is likely that the past reconstruction segment contains a better replica of the beginning of the lost packet than does the future reconstruction segment, and that the future segment contains a better estimate of the end of the lost packet than does the past segment. Therefore, the $L+2P$ weights for the past segment are

$$W_p(k) = \frac{L+2P-k}{L+2P-1}\,, \quad k = 1,2,\ldots,2P+L \tag{6}$$

and the weights for the future segment are

$$W_f(k) = 1 - W_p(k)\,. \tag{7}$$

The two segments are multiplied by these weights, and added together.

Since the energy of a speech signal changes continuously, it is advisable to normalize the amplitude of the reconstructed packet to ensure that it is not much louder or quieter than the speech surrounding it. This normalization was attempted in a few different ways. In each case I multiplied the samples in the reconstruction packet by a constant in order to make the amplitude of this packet equal to that of the packet preceding the missing one.

I experimented with three amplitude measures. Denoting the samples in a received packet or a packet-length reconstruction segment by $z(1),z(2),\ldots,z(L)$, the amplitude measures are the square root of the energy,

$$A_1 = \sqrt{\sum_{l=1}^{L}[z(k)]^2}\,, \tag{8}$$

the sum of absolute magnitudes,

$$A_2 = \sum_{l=1}^{L}|z(k)|\,, \tag{9}$$

and the peak-to-peak amplitude,

$$A_3 = max[z(k)] - min[z(k)]\,. \tag{10}$$

Doing the same type of experiments as before, I found the quality of the output speech to be more sensitive to the method of normalization than to the pattern-matching measure. For the two-sided scheme, the energy measure, Equation (8), gives the best results. For the one-sided scheme, the sum of absolute magnitudes, Equation (9), is as effective as the energy measure. Neither scheme works well with peak-to-peak amplitudes.

### *2.1.4 Merging the Reconstruction Packet with the Received Speech*

Next, the packet is inserted into the stream of received speech by merging the merge window portions at its ends with the speech surrounding the missing packet. This merging is done with raised cosine weighting. The $P$ weights for the last $P$ samples of the packet received before the missing one are

$$W_1(k) = \frac{1}{2}\left[1+cos\left(\frac{\pi(k-1)}{P-1}\right)\right], \quad k = 1,2,..,P \tag{11}$$

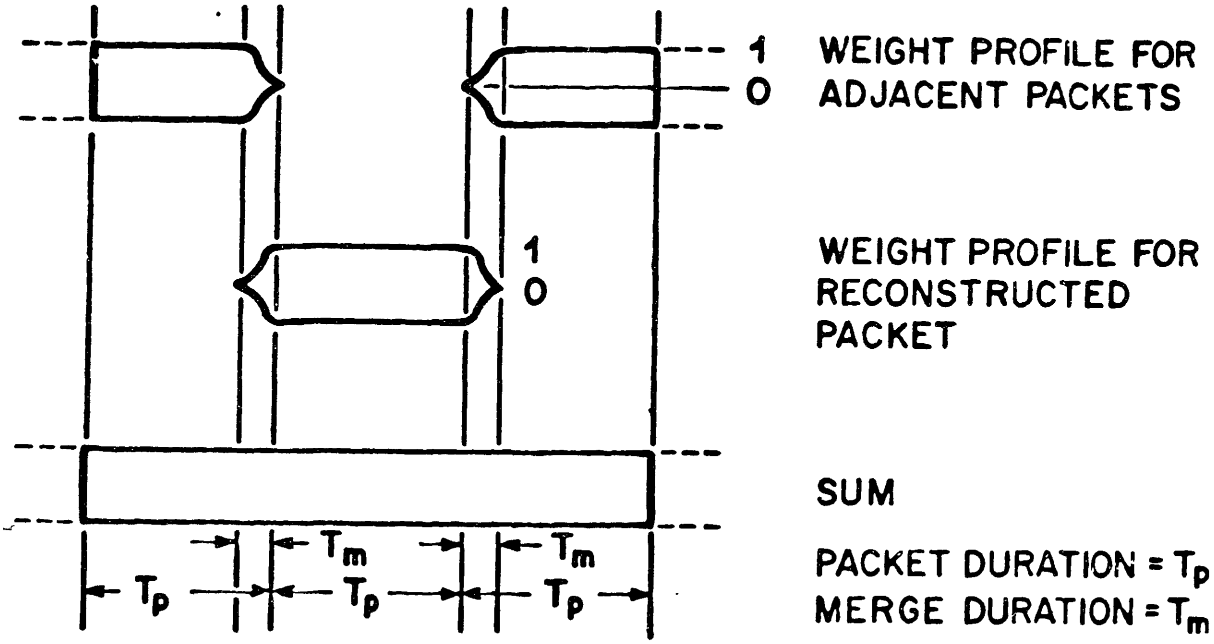
and the $P$ weights for the first $P$ samples of the reconstruction packet are

$$W_2(k) = 1 - W_1(k)\,. \tag{12}$$

Naturally, it follows that the $P$ weights for the last $P$ samples of the reconstruction packet are $W_1(k)$, and that the $P$ weights for the first $P$ samples of the packet received after the missing one are $W_2(k)$.

The merge segments are multiplied by their respective weights and added together, and the $L$ remaining samples of the reconstruction packet fit neatly between them. Figure 6 illustrates how the merging works.

**Figure 6.** Merging a reconstructed packet using raised cosine weighting.



### *2.1.5 The One-Sided Scheme*

The one-sided scheme simply replaces the missing packet with the past reconstruction segment, still normalizing the segment and merging its ends. It introduces less delay and requires less memory than the two-sided scheme. Subjective tests reveal that this decrease of computation and memory does not compromise speech quality. (Section 4.1)

### *2.1.6 Multiple Lost Packets*

When two packets in a row are missing, the two-sided scheme will treat them as if they are one double-sized missing packet, and reconstruct them as such. The templates, search windows, and merge windows remain the same length, but twice as much speech following (or preceding) the match is used in the reconstruction. Of course, the search window in this case begins two packets plus a merge window $(2L+P)$ from the missing one, instead of only one. When more than two packets in a row are missing, I adopt the same approach. Note that the one-sided scheme does not change at all when two or more packets in a row are missing.

When a packet within the search window is missing, the two-sided scheme sees it as a silent packet and chooses a match accordingly. Such a situation can occur only in the future search window, as any missing packets in the past search window would already have been reconstructed. Therefore this situation also has no effect on the one-sided scheme.

### *2.1.7 Experimental Results*

I have investigated the effects of the packet replacement schemes on 11.15 seconds of speech spoken by two women and two men, sampled at 8 kHz, and processed by a quantizer with $\mu = 255$ companding. The missing packets were independently selected by a uniform random number generator. I have conducted the investigations by listening to the speech processed by the one-sided and two-sided algorithms, using various combinations of packet size, template size, search window size, and missing packet ratio. In this investigation, I did not include the merge window in order to merge the reconstruction packets with the received speech. The merging technique was proposed in a subsequent study conducted by Lockhart and Goodman, and reported in a paper by Goodman, Jaffe, Lockhart, and Wong. [7] My purpose in these investigations has been to evaluate the effects of system parameters and the missing packet ratio on the quality of the reconstructed speech.

In Section 2.1.2, I defined several matching criteria and methods of normalization, and discussed their influence on the quality of the reconstruction scheme. I now present the effects of packet length ($L$), template length ($M$), and search window length ($N$) on signal-to-noise ratio and on my perceptions of speech quality. To compute the signal-to-noise ratio ($SNR$), I let $x[n]$ be the samples of the companded speech without any dropped packets, and I let $y[n]$ be the samples of the companded speech after packets have been dropped and reconstructed. Thus the $SNR$ does not include quantization noise. The formula for computing the total $SNR$ is

$$SNR = 10\log\left[\frac{\sum\limits_n (x[n])^2}{\sum\limits_n (y[n]-x[n])^2}\right]. \tag{13}$$

A reference to the average $SNR$ per missing packet indicates that an $SNR$ is computed as in Equation (13) for each missing packet, and then all of these $SNR$'s are averaged.

In addition, I will quote the results of the study which examined merge window length ($P$). In performing these evaluations, I have used the magnitude difference measure, Equation (4), for pattern matching, and the square-root energy normalization, Equation (8), of reconstruction segments.

*2.1.7.1 Packet Size* For a given fraction of packets missing, the packet size has a strong effect on the perceived nature of the reconstructed speech. (Section 1, Table 1) With very small packets (1 or 2 ms, $L$ = 8 or 16), with 10 percent of the packets missing, there is a constant. annoying crackle. For very large packets (32 ms or more), the speech sounds as though the person is trying to gargle while speaking. For sizes in between, the crackles become pops, and occur infrequently, rather than constantly like the crackle. To my ears, the packet size most tolerant to packet loss is 8 ms (64 samples), although 16 ms is also good. Signal-to-noise ratio of reconstructed speech is not a good indication of how the quality changes with packet size; it improves as the packets get smaller and smaller.

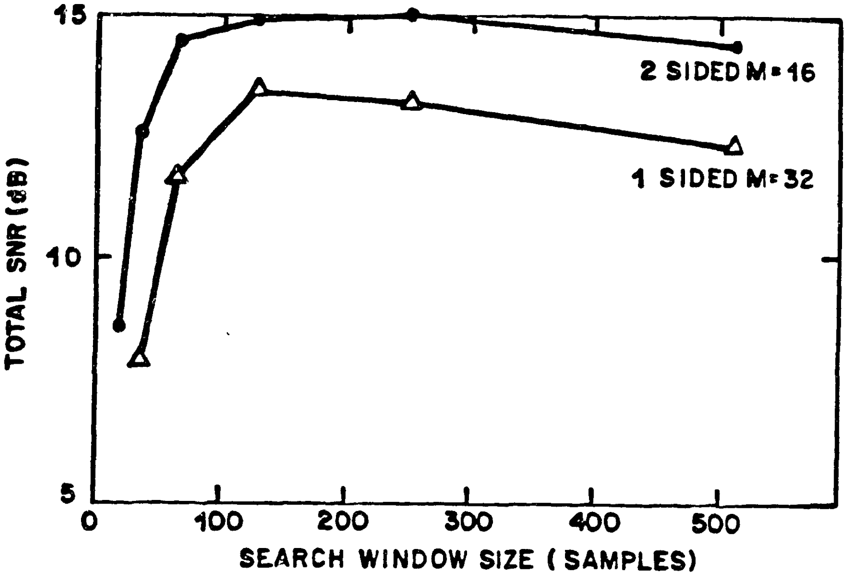
*2.1.7.2 Search Window* There is an optimum search window duration. If the search window is too short it omits the best reconstruction waveform. If it is too long it contains speech that is unrelated to the missing packet. Nevertheless, there is a chance that a small segment of this speech is well matched to the $M$ samples in the template, a situation which can result in the selection of a suboptimum reconstruction segment. In my experiments I found that regardless of packet size, the one-sided scheme worked best with a 16 ms search window, and that 8 ms was best with the two-sided scheme. Speech quality does not deteriorate appreciably when the search window is longer than optimum.

I observed that signal-to-noise ratio is a reasonably good indicator of the relationship of perceived quality to search window duration. Figure 7 shows the total $SNR$ as defined in Equation (13) (measured across 11 seconds of speech) as a function of the number of samples in the search window. Figure 8 displays the average $SNR$ (in dB) per missing packet, a measure found in a previous study to be indicative of the relative quality of reconstruction methods. [4] Figures 7 and 8 have 64 samples per packet and the missing packet ratio is 9.2 percent. In the one-sided measurements the template contains $M = 32$ samples. In the two-sided scheme $M = 16$.

Note in each curve in Figures 7 and 8 that the shortest search window is equal in duration to the template (16 or 32 samples). In this case the algorithm replaces the missing packet with the previous packet (one-sided) or a weighted average of the packet that precedes the missing packet and the packet that follows it (two-sided).

*2.1.7.3 Template* As in the case of the search window, there is a minimum acceptable template duration. If $M$ is too small, there is simply insufficient speech information. The quality also goes down if the template is too long. Again, the best size appears to be independent of the packet size. I found that 2 ms (16 samples) was the best template for the two-sided scheme, and 4 ms was best for the one-sided scheme.

**Figure 7.** Total signal-to-noise ratio as a function of search window size. There are 64 samples per packet and 9.2 percent of the packets are missing. In the one-sided version, $M = 16$ samples per template; $M = 32$ with two-sided reconstruction.



*SNR* correlates well with the effect of template duration on perceived quality, in that the relative *SNR* measures generally conform to my impressions of relative quality. Figure 9 shows the average *SNR* per missing packet as a function of template size.

**Figure 8.** Average signal-to-noise ratio per missing packet for the same conditions as Figure 7.

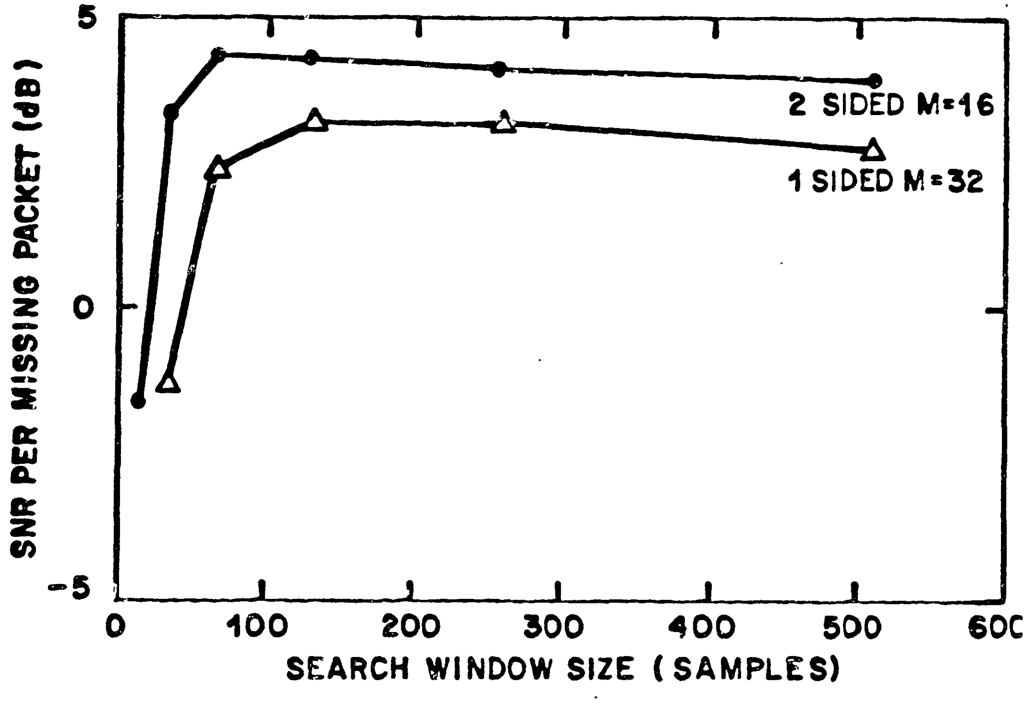
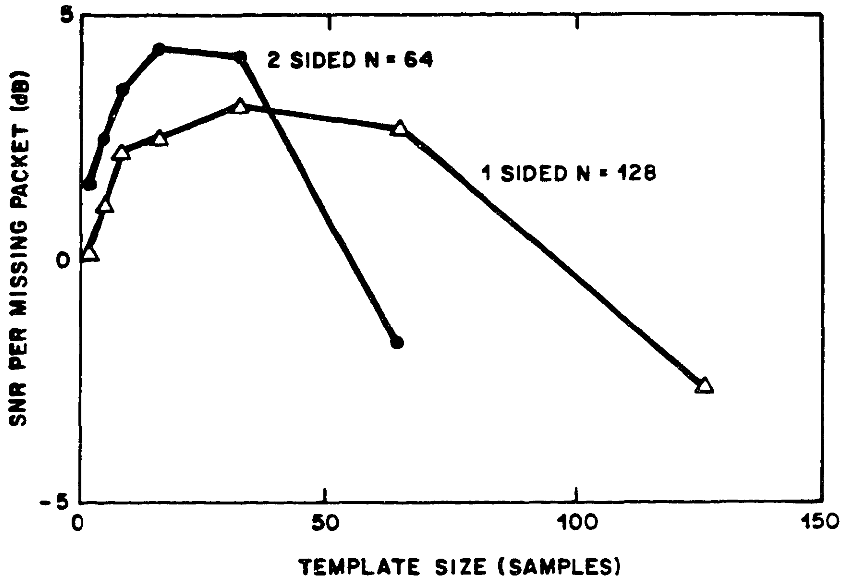


**Figure 9.** Average signal-to-noise ratio per missing packet as a function of template size. There are 64 samples per packet and 9.2 percent of the packets are missing. With one-sided pattern matching there are $N = 128$ samples in the search window; $N = 64$ in the two-sided case.



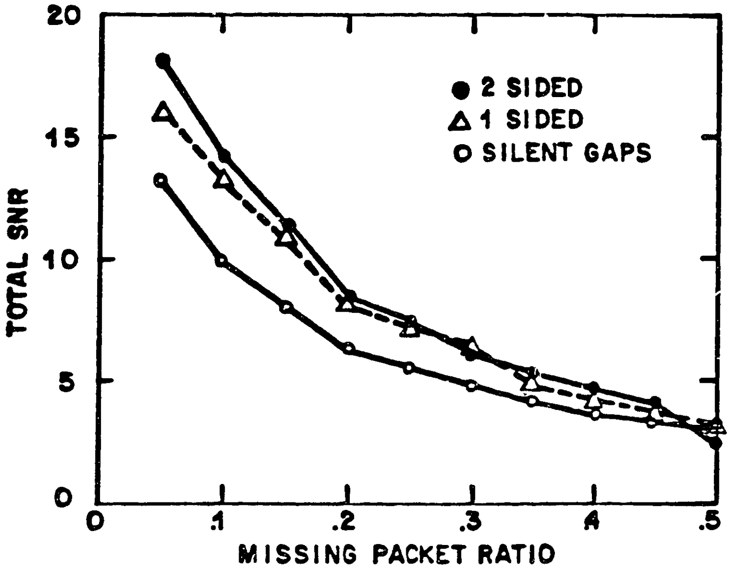
*2.1.7.4 Merge Window* A study done by Lockhart and Goodman [7] indicates that a merge-window duration of 1 ms yields good results.

*2.1.7.5 Missing Packet Ratio* As expected, the *SNR* and the perceived quality decline as the fraction of packets missing increases. Figure 10 shows the dependence of total *SNR* on missing packet ratio for both versions of my packet recovery method and for missing packets replaced by silent gaps. My listening experience suggests that communication breaks down when more than 30 percent of the speech is lost. When only half of the packets arrive, the two-sided scheme yields badly garbled speech, and the one-sided scheme yields speech interspersed with beeps and chirps very like the voice of the robot R2D2 in the movie 'Star Wars.' This is because when many packets in a row are lost, the one-sided scheme repeats the same segment, creating a highly periodic signal.

### *2.1.8 A Pitch Detection Method*

In this section I describe another algorithm which Lockhart and Goodman used on PCM speech. [7] It is also a waveform substitution technique, and therefore I compared my

**Figure 10.** Total *SNR* as a function of missing packet ratio for both versions of the reconstruction scheme and for missing packets replaced by gaps of silence.



algorithms to it in a subjective test.

The pitch detection method is simply to detect the pitch period ($T$ samples) of the speech immediately preceding the missing packet, and then to replicate the last $T$ samples of the received speech for the length of the missing packet. Of course, the edges of the new packet are merged with the surrounding speech.

The pitch detector consists of two peak detectors, one positive and one negative. They each remember the positions of the last three significant peaks they detected, so that at any given moment, four estimates of pitch period are available. Since voiced speech usually has higher energy than unvoiced speech, center clipping the signal helps to determine the voicing. The actual pitch period used depends on the reliability of the various estimates. For instance if the last significant peak was too long ago, one can assume the speech preceding the missing packet was unvoiced. Conversely, if the resultant pitch estimate is too high, perhaps one of the pitch detectors detected more that one significant peak per period. When the pitch estimate is unreliable, one simply repeats the previous packet.

## 2.2 Algorithms for ADPCM Speech

In my studies of dropped packets of ADPCM encoded speech, I have used the CCITT (International Consultative Committee on Telephony and Telegraphy) standardized 32kbit/s ADPCM without tandeming [8], rather than the packet-wise adaptive version used by Jayant and Christensen (Section 1). This means that the step size and the predictor change sample by sample, and that these parameters are not transmitted over the channel. The encoder takes in a PCM signal and outputs an ADPCM signal, while the decoder takes in an ADPCM signal and outputs a PCM signal.

**Figure 11.** block diagram of the ADPCM coder

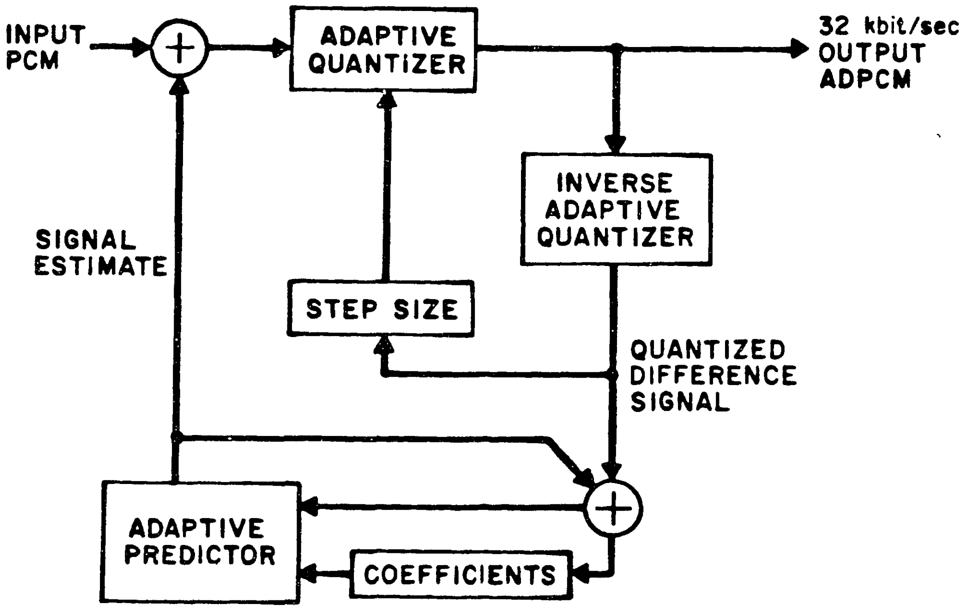


Figure 11 displays a block diagram of the ADPCM coder, and Figure 12 displays a block diagram of the ADPCM decoder. The most serious obstacle to good reconstruction of missing packets is that the encoder and decoder lose synchronization. Therefore, if one packet is dropped, it may cause fifteen or twenty subsequent packets to be corrupted at the receiver.

### *2.2.1 Overview of Various Algorithms*

The presence of a decoder in the receiver opens several options for reconstruction algorithms. I have developed and simulated them all in the C Programming Language. Each algorithm is a

**Figure 12.** block diagram of the ADPCM decoder

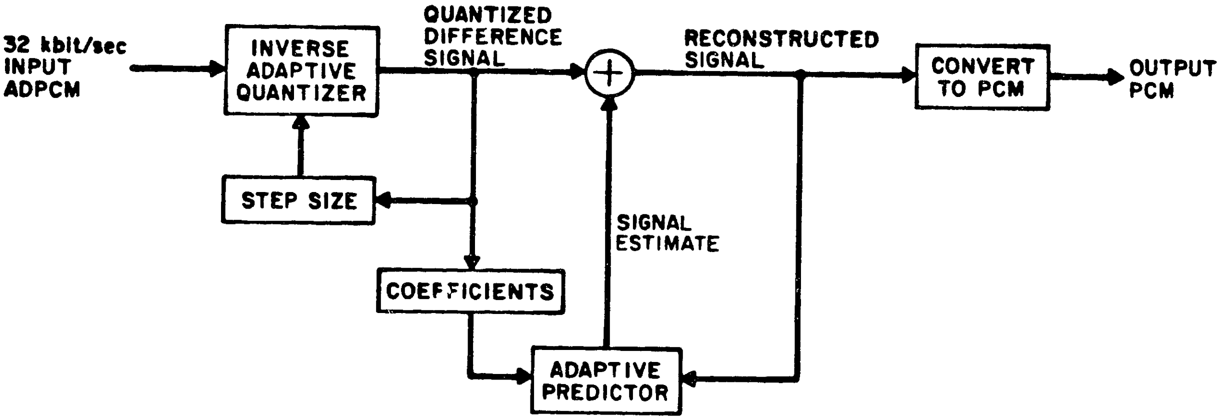
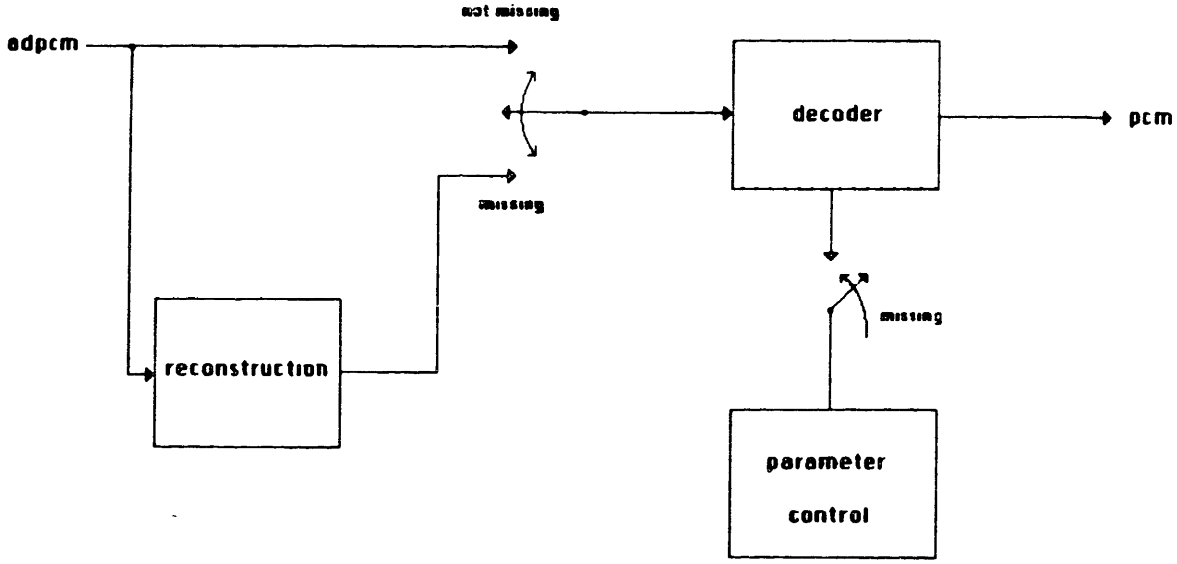


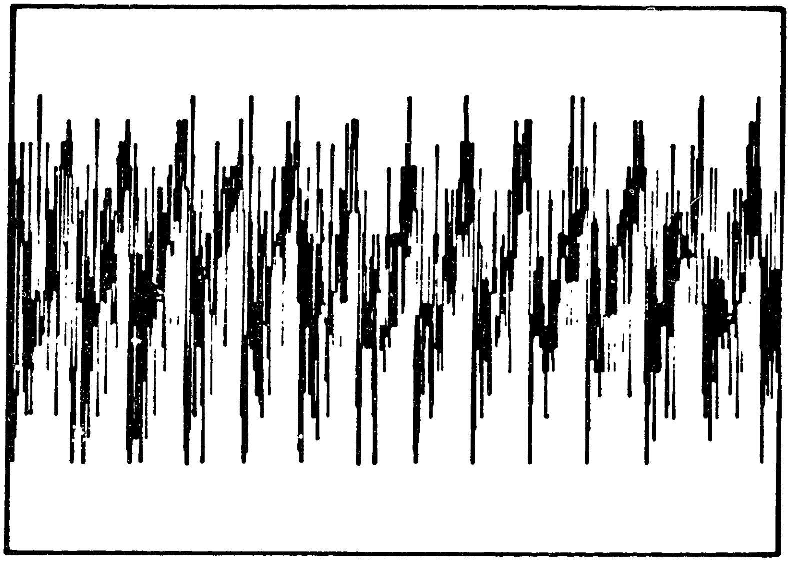
**Figure 13.** reconstruction in the ADPCM environment



combination of three factors: reconstruction environment, reconstruction technique, and treatment of the decoder's parameters.

*2.2.1.1 Reconstruction Environment* Reconstruction can be done either on the ADPCM signal before it is decoded, or on the PCM signal after it is decoded. Figure 13 shows reconstruction in the ADPCM environment, while Figure 14 shows a segment of ADPCM speech. Each ADPCM

**Figure 14.** ADPCM speech



sample has one of 16 possible values, but the meaning of that value depends upon the recent history of the ADPCM signal. Because of this, the ADPCM signal is by no means continuous and quasiperiodic like speech. Nevertheless, I have used some of the same techniques on the ADPCM signal as I have on the PCM signal.

**Figure 15.** reconstruction in the PCM environment

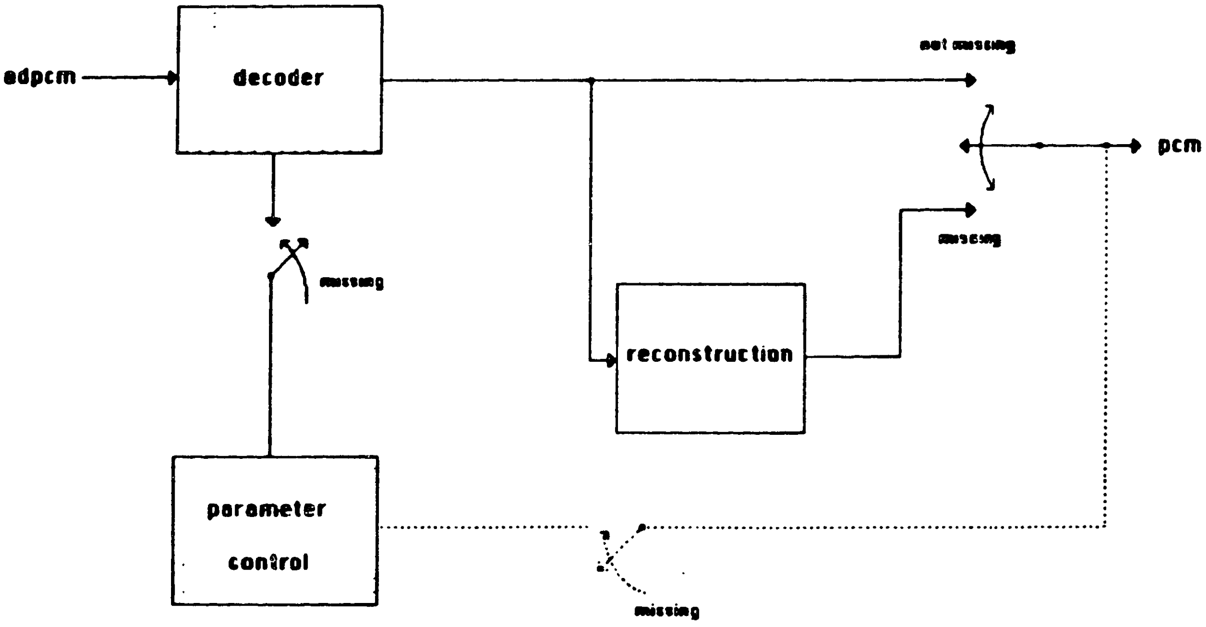
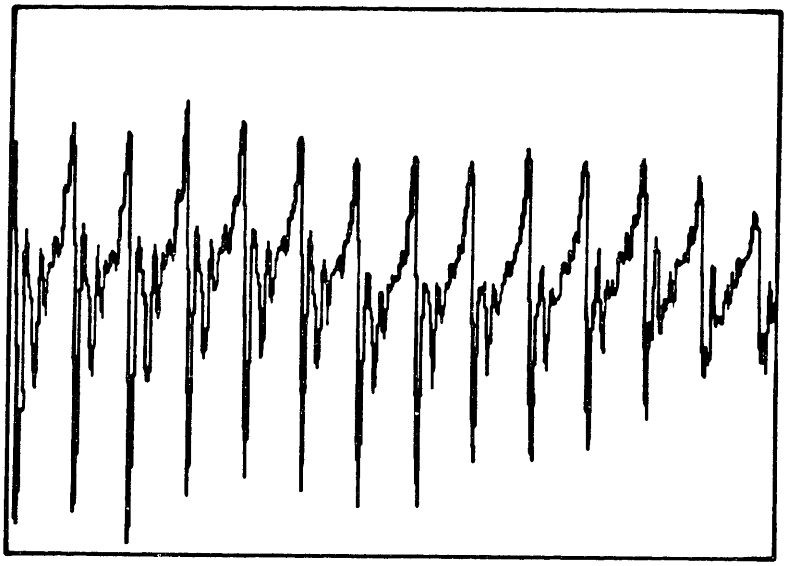


**Figure 16.** PCM speech



Reconstruction in the PCM environment is displayed in Figure 15, while PCM speech is displayed in Figure 16. This is the PCM version of the same segment of speech as in Figure 14. Even though the two signals contain some of the same information, the speech periodicity is better revealed by the PCM waveform and more amenable to analysis by pattern matching. Therefore, a missing packet can be reconstructed the same way as in Section 2.1. The question arises, however, as to what will become of the speech following the missing packet, since the decoder has missed some input.

*2.2.1.2 Reconstruction Technique* Since things can get very complicated when dealing with an adaptive quantizer and an adaptive predictor, I have tried some "first order" reconstruction schemes, along with more computationally intensive schemes, in both reconstruction environments. The simplest technique is to replace the missing packet with silence. This involves plugging sample values '0' into a PCM packet, or alternating sample values '-1' and '+1' in an ADPCM packet.

For another very simple technique I repeat the previous packet in place of the missing one. In the ADPCM environment, I do not merge the ends of the replacement packet with the surrounding signal, because there seems to be no smoothness constraint on the ADPCM signal.

However, I do merge the ends of the packets in the PCM environment.

The third reconstruction scheme is the one-sided scheme which I have used for PCM speech. (Section 2.1.5) I have tried it in both reconstruction environments, merging as before.

*2.2.1.3 Decoder Parameters* Whether in the PCM or ADPCM environment, no matter what the reconstruction scheme, the question arises as to what to do with the quantizer step size and predictor coefficients when a packet is missing. I have tried three things: (1) reset them to the decoder start-up state, (2) freeze them so that they do not change at all when a packet is missing, (3) update them in some way.

It is impossible to freeze the parameters in the ADPCM environment, because the replacement ADPCM packet must be sent through the decoder, forcing the decoder parameters to change. Updating in this environment consists of simply sending the replacement packet through the decoder without resetting its parameters.

In the PCM environment, freezing the parameters means that when a missing packet is encountered, it is not sent through the decoder. Instead, some replacement PCM packet (perhaps past speech as in the one-sided scheme) is played out, and then the packet following the missing one is sent through the decoder. Thus the parameters have not been changed by the missing packet. When updating the parameters in the PCM environment, I employ an encoder at the packet receiver. When good packets arrive, the encoder parameters are the same as decoder parameters. When there is a missing packet, it is not sent through the decoder, but instead is reconstructed as a PCM packet. Then these PCM samples are sent to the encoder, thus updating the encoder parameters. After the missing packet interval, the decoder parameters are set to equal the encoder parameters. Again, the packet following the missing one is decoded as usual.

### *2.2.2 Experimental Results*

I tested all of these algorithms on two sources of speech, each ten seconds long, one spoken by a man, the other spoken by a woman. The speech was 8 kHz, 16 bit linear PCM. As before, the missing packets were selected by a uniform random number generator. All of the algorithms used 16 ms packets. The pattern matching technique used Equation (4) for pattern matching, Equation (8) for normalization, a 16 ms search window, a 4 ms template, and a 1 ms merge window, for reasons explained in Section 2.1 of this paper. My intent was to discover which algorithms were good enough to merit formal subjective testing.

*2.2.2.1 Reconstruction Environment* As it turns out, one should not do this kind of reconstruction on the ADPCM side of the decoder, as it results in loud pops which are painful if not dangerous to the listener. These pops are the result of the step-size and predictor coefficients of the decoder diverging from those of the encoder.

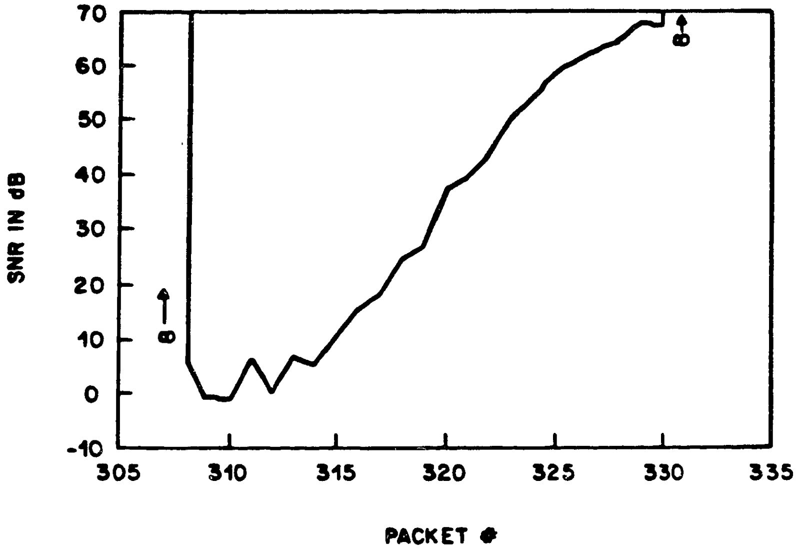
Reconstruction on the PCM side does not yield such hazards, probably because one never decodes anything except original ADPCM signal. Even when decoder parameters are updated, a reconstruction PCM packet is encoded merely for a reading of the parameters, as opposed to trying to decode reconstructed ADPCM signal.

*2.2.2.2 Reconstruction Technique* To my ears, substituting with silence was the worst of the three techniques. Surprisingly, I could not tell the difference between repeating the previous packet and pattern matching.

*2.2.2.3 Decoder Parameters* After listening, I judge that it is better to freeze or update the parameters, rather than reset them to the start up state. This is not surprising, since the start up state is meant to be the best state for the beginning of a speech burst, but the missing packets often occur in the middle of a burst. Therefore, it is likely that frozen or updated parameters more closely approximate the true parameters than would reset parameters.

*2.2.2.4 Propagation of Errors* Figure 17 shows the *SNR* per packet vs. packet number, beginning just before a typical missing packet. The *SNR* does not include quantizing noise. The algorithm used was repeating the previous packet and freezing the decoder parameters. Notice that the quality degrades slightly one packet before the missing one, because the merging process distorts the last millisecond of the packet.

**Figure 17.** *SNR* per packet vs. packet #, starting just before a typical missing packet. The algorithm used was repeating the previous packet and freezing the decoder parameters.



### 3. Description of Subjective Tests

The purpose of the subjective tests is to examine the relative merits of the various reconstruction techniques for PCM and ADPCM speech. The first subjective test also examines the extent to which merging the reconstructed packet ends into the received speech contributes to the quality of the output speech.

### 3.1 Source Speech

In the subjective tests, I used source speech that had been prepared for other subjective tests. It consists of two different sets of three Harvard sentences. Each set was spoken by two male and two female speakers, for a total of eight different sets of sentences. The speech was recorded in a sound-proof chamber with a Sony ECM-220T electret condenser microphone, and digitized through a DSC-200 Digital Audio Data Conversion System (Digital Sound Corporation). Then it was put through a 3.2 kHz low-pass filter and sampled at 8 kHz. The quantizer was a sixteen-bit linear quantizer. After digitization, the speech was equalized for active speech power using a British Telecom SV6 Speech Voltmeter so that there was only a 0.3dBm range among the various stimuli.

### 3.2 Testing Facilities

The subjects sat in partitioned cubicles in a room with acoustic material on the walls. The background noise level was 35dBA. They listened to the speech stimuli over calibrated handsets. The amplitude of the speech at the handsets was -27.9 dBm for the PCM test, and -28.5 dBm for the ADPCM test. Three seconds of silence preceded each stimulus, and the subjects had five seconds to respond to each stimulus. They responded by pushing one of five buttons labelled "Excellent", "Good", "Fair", "Poor", and "Unsatisfactory." Three colored lights on the same panel indicated when the subjects should "WAIT", "LISTEN", or "VOTE." A bell at the experimenter's terminal indicated if a subject did not respond within five seconds. When all of the subjects responded, or if the five seconds was up, the ratings were recorded on a diskette, and the next trial was presented.

The following describes the testing procedure. First, instructions were read to the subjects. A copy of these instructions is in Appendix A at the end of this paper. Then twelve practice trials were presented to the subjects. (Ten practice trials in the case of the ADPCM test.) These practice trials covered the full range of the quality of the various stimuli. Next, the subjects responded to half of the experimental trials, and then they took a break during which a snack was served. After the break, the subjects responded to the other half of the experimental trials.

### 3.3 Testing the Algorithms for PCM Speech

In the first subjective test, I compared the relative quality of the two pattern matching schemes (two-sided and one-sided), the pitch detection scheme, the method of simply repeating the previous packet, and the simplest approach of all -- substituting silence for the missing packet. I also wanted to determine whether merging improved the quality significantly. Therefore, the first subjective test included the two pattern matching schemes, the pitch detection scheme, and repetition of the previous packet, both with and without merging, and in addition, the silence substitution method without merging. I used a packet size of 16 ms in every process. In the processes with merging, I used a merge window size of 1 ms. The pattern matching schemes used a template of 4 ms and a 16 ms search window. They also used Equation (4) for pattern matching, and Equation (8) for normalization.

In order to enable the reader to compare my results to results of published tests, I included reference noise conditions. The reference noise was produced by adding to each sample a value proportional to its amplitude and randomly positive or negative. The constant of proportionality depends upon the signal to noise ratio desired. Specifically, if $c$ is the constant of proportionality, and $s$ is the desired signal to noise ratio in dB, their relation is:

$$s = 20log\frac{1}{c} \tag{14}$$

$$c = 10^{-\frac{s}{20}} \tag{15}$$

This modulated noise reference process is commonly used in subjective tests of digital speech communication techniques.

In order to find out how quickly the various processes degrade with increasing missing packet ratio, and whether one process might be better for low ratios and another better for high ratios, I tested all of the processes with missing packet ratios of two, four, eight, and sixteen percent. To cover the full range of quality, I used signal-to-noise ratios of six, fourteen, twenty-two, and thirty decibels in the reference noise conditions. So, I tested a total of 320 different conditions. The following ten processes were used to process speech from eight speakers, at four missing packet ratios (or signal-to-noise ratio in the case of the reference noise):

(1) modulated noise reference

(2) one-sided pattern matching with merging

(3) one-sided pattern matching without merging

(4) pitch detection method with merging

(5) pitch detection method without merging

(6) repeating previous packet with merging

(7) repeating previous packet without merging

(8) substituting silence without merging

(9) two-sided pattern matching with merging

(10) two-sided pattern matching without merging

This was approximately 53.5 minutes of speech.

In a given speech file, the packets to drop and reconstruct are chosen by a random number generator. The quality of the reconstructed speech depends upon which packets were dropped, because some are harder to reconstruct than others. In order for this effect to average out over the course of the test, each of the 320 stimuli was produced with a different random seed.

The subjects' opinions of the stimuli is also dependent upon the order in which they are presented. That is, a subject might vote higher for a condition occurring after a very low quality stimulus than for the same condition occurring after a high quality stimulus. In order to average out this effect, two versions of the test were given, with two different permutations of the stimuli, and two different sets of subjects. The random seeds which chose the missing packets were different in all 640 conditions.

The two versions of this subjective test were administered on November 14, 1985 at 9:00 am and at 1:00 pm to a total of 21 women, eleven in the morning and ten in the afternoon. All of the women were housewives, most of whom had participated in multiple listening tests in the past. Their ages ranged from 20 to 60 years, and their median age was 48 years.

### 3.4 Testing the Algorithms for ADPCM Speech

In the second subjective test, I compared the relative quality of the various combinations of reconstruction scheme and decoder parameter control, with missing speech reconstructed on the PCM side of the decoder. The reconstruction techniques are substituting silence, repeating the previous packet with merging, and one-sided pattern matching with merging. The parameter control methods are freezing, resetting, and updating the decoder parameters. Since silence substitution is included only as a no-cost baseline, I did not experiment with parameter control. Instead, I simply chose to freeze the decoder parameters when substituting silence, because freezing the parameters emulates what would actually happen if a packet were dropped in a network implementing speech activity detection and the CCITT standard ADPCM. [8] In this situation, the parameters would not be reset unless approximately ten consecutive packets were missing or silent. The pattern matching technique was identical to the one-sided scheme used in the PCM subjective test. I merged the reconstruction packet with the surrounding speech in every case except silence substitution.

The effect of missing packets is more severe with ADPCM coding than PCM coding. To cover approximately the same quality range as in the PCM test, I used a lower set of missing packet ratios in the ADPCM test: one, two, four, and eight percent. The same reference conditions as before covered this range of quality.

There were 256 conditions to be tested, for a total of 42.7 minutes of speech. The following eight processes were used to process speech from eight speakers, at four missing packet ratios (or signal-to-noise ratios):

(1) modulated noise reference

(2) pattern matching with merging, freezing decoder parameters

(3) pattern matching with merging, resetting decoder parameters

(4) pattern matching with merging, updating decoder parameters

(5) repeating previous packet with merging, freezing decoder parameters

(6) repeating previous packet with merging, resetting decoder parameters

(7) repeating previous packet with merging, updating decoder parameters

(8) substituting silence without merging, freezing decoder parameters

Again, I ran two versions of the test, each with a different presentation order and different patterns of missing packets in each of the 512 conditions. The two versions of this subjective test were administered on December 4, 1985 at 9:00 am and 1:00 pm to a total of 22 women, eleven at each session. All of the women were housewives, most of whom had participated in multiple listening tests in the past. Their ages ranged from 20 to 60 years, and their median age was 43 years.

## 4. Results of Subjective Tests

I analyzed the data from the tests on a VAX 11/750 computer under a *UNIX*$^{TM}$ operating system, using the S language for data analysis. [9] The detailed statistical analysis of the results is in Appendix B at the end of this paper. In this section I present and interpret the results of the analysis.

### 4.1 Results for the PCM Algorithms

I made three hypotheses before analyzing the data. In the first place, I expected that it would be possible to combine both versions of the test as samples taken from the same population. This is explored further in Appendix B. In the second place, I expected that any process with merging would be significantly better than the same process without merging. Finally, I expected pattern matching to be significantly better than the other methods, silence substitution significantly worse, with pitch detection and repeating the previous packet comparable to each other. The second and third hypotheses were based on the listening experiences of myself and a couple of colleagues.

Table 4 shows, for each missing packet ratio, which processes had mean opinion scores (MOS's) which were significantly higher than those for other processes. Table 3 assigns a two-letter name to each process for the purpose of making Table 4 concise.

**TABLE 3**

| Name | Process |
|---|---|
| OM | one-sided pattern matching with merging |
| ON | one-sided pattern matching without merging |
| PM | pitch detection method with merging |
| PN | pitch detection method without merging |
| RM | repeating previous packet with merging |
| RN | repeating previous packet without merging |
| SN | substituting silence without merging |
| TM | two-sided pattern matching with merging |
| TN | two-sided pattern matching without merging |

**TABLE 4**

| Missing Packet Ratio | On each line, processes on left are significantly better than processes on right. | |
|---|---|---|
| 2% | PM | RM,RN,SN |
| | OM,PN,TM,TN,ON | RN,SN |
| | RM,RN | SN |
| 4% | PM,PN | TN,OM,RN,RM,SN |
| | TM | OM,RN,RM,SN |
| | ON | RN,RM,SN |
| | TN | RM,SN |
| | OM,RN,RM | SN |
| 8% | PM,PN | TM,OM,TN,ON,RM,RN,SN |
| | TM,OM,TN,ON | RM,RN,SN |
| | RM,RN | SN |
| 16% | PM,PN | OM,ON,TN,TM,RM,RN,SN |
| | OM,ON | RN,SN |
| | TN,TM,RM,RN | SN |

The results indicate that merging the packet ends does not contribute significantly to quality at any missing packet ratio. This is surprising, since merging sounded better to myself and my colleagues. Apparently, the subjects did not notice this difference to a significant degree.

It seems that substituting silence for the missing packet produces the worst quality of speech, while the pitch detection method produces the best. It was expected that substituting silence would produce the worst quality, as it guarantees a discontuity of the signal at both ends of the packet. Pitch detection is the best method to use, because as shown in Table 4, it is usually significantly better than the other methods. Furthermore, some implementations of the simple pitch detection algorithm used by Lockhart and Goodman [7] require no more computation than pattern matching for this improved quality.

In some cases, pattern matching is better than repetition of the previous packet, but in other cases they seem to be of somewhat comparable quality. I consider one-sided pattern matching to be superior to two-sided pattern matching, because it produces the same quality of speech without the delay of waiting for the next two packets after the missing one before reconstruction can begin.

Figures 18-24 show plots of the MOS scores of various processes at various missing packet ratios. Figure 18 shows the MOS's for the reference noise process at each SNR. Figures 19-22 each show the MOS's of a process with merging and without merging at each missing packet ratio. Notice in all of them that the MOS's for a given process at a given missing packet ratio do not change significantly in the absence of merging.

**Figure 18.** MOS vs. *SNR* for reference noise process

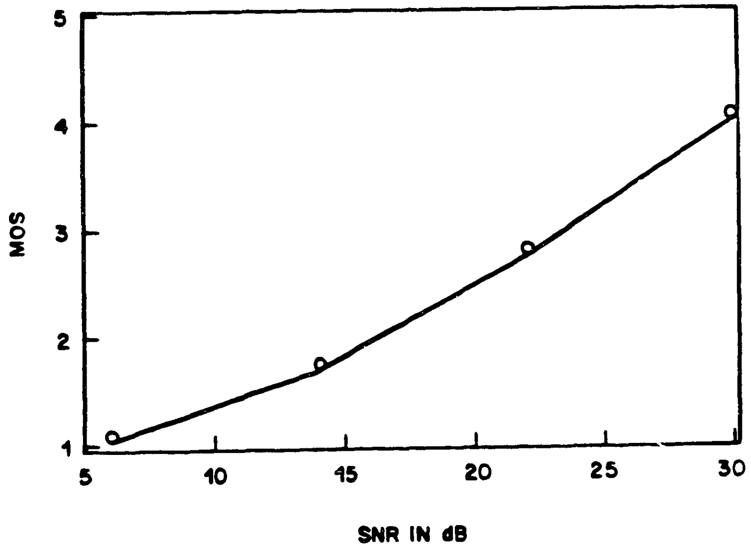


**Figure 19.** MOS vs. missing packet ratio for one-sided pattern matching with (OM) and without (ON) merging: O = OM and X = ON

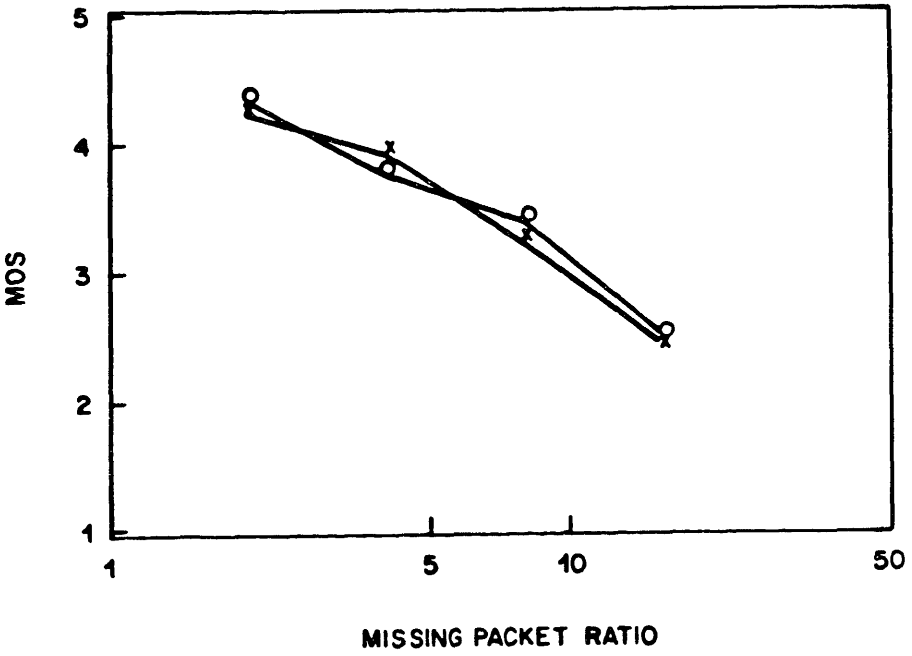


**Figure 20.** MOS vs. missing packet ratio for pitch detection method with (PM) and without (PN) merging: O = PM and X = PN

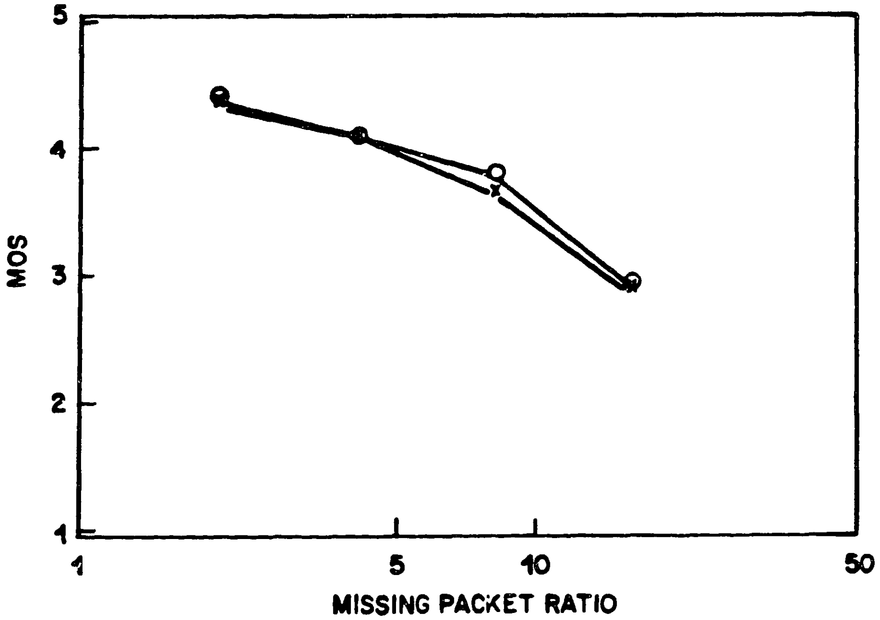
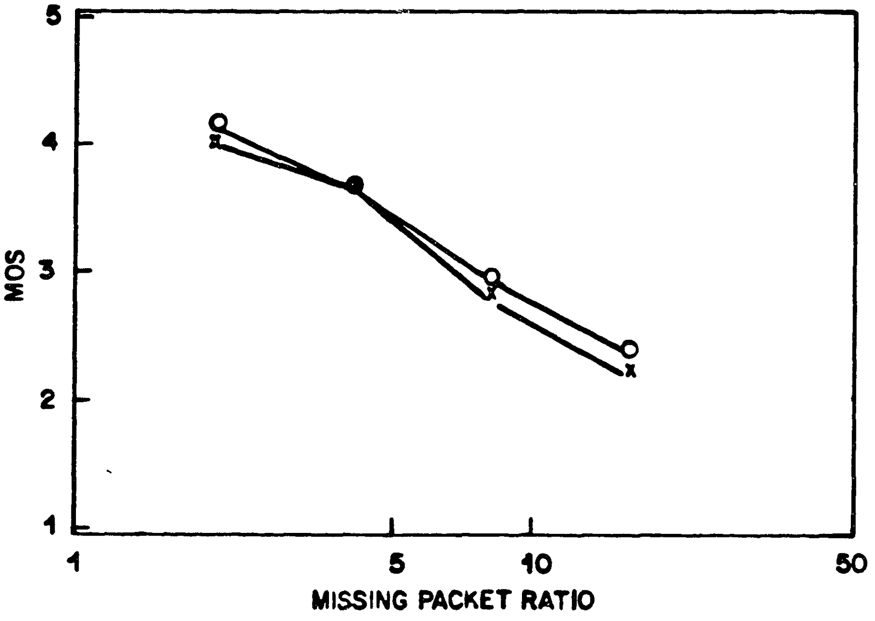


**Figure 21.** MOS vs. missing packet ratio for repeating the previous packet with (RM) and without (RN) merging: O = RM and X = RN



Since the merging did not improve the processes significantly, I pooled the results for each process with and without merging, and put the results in Table 6 according to the key in Table 5.

**Figure 22.** MOS vs. missing packet ratio for two-sided pattern matching with (TM) and without (TN) merging: O = TM and X = TN

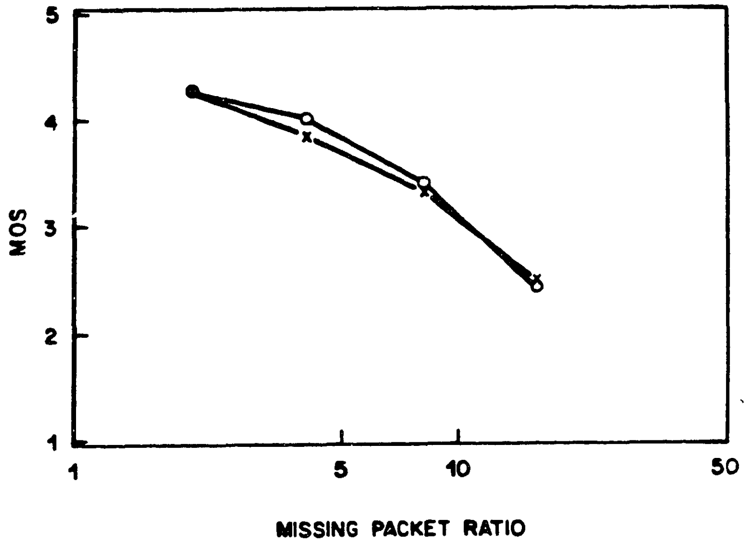


**TABLE 5**

| Name | Process |
|---|---|
| O | one-sided pattern matching |
| P | pitch detection method |
| R | repeating previous packet |
| S | substituting silence |
| T | two-sided pattern matching |

**TABLE 6**

| Missing Packet Rate | On each line, processes on left are significantly better than processes on right. | |
|---|---|---|
| 2% | P,O,T | R,S |
| | R | S |
| 4% | P | T,O,R,S |
| | T,O | R,S |
| | R | S |
| 8% | P | T,O,R,S |
| | T,O | R,S |
| | R | S |
| 16% | P | O,T,R,S |
| | O | R,S |
| | T,R | S |

Figure 23 shows MOS vs. missing packet ratio for these results, and Figure 24 shows a scatter plot of Figure 23, but with ellipses drawn on it. Each process' MOS is significantly different

**Figure 23.** MOS vs. missing packet ratio for each process: see Table 5 for key to symbols

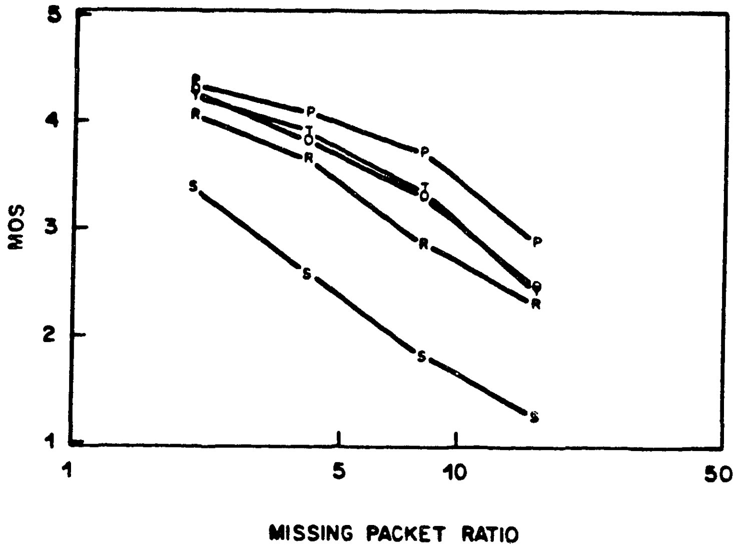
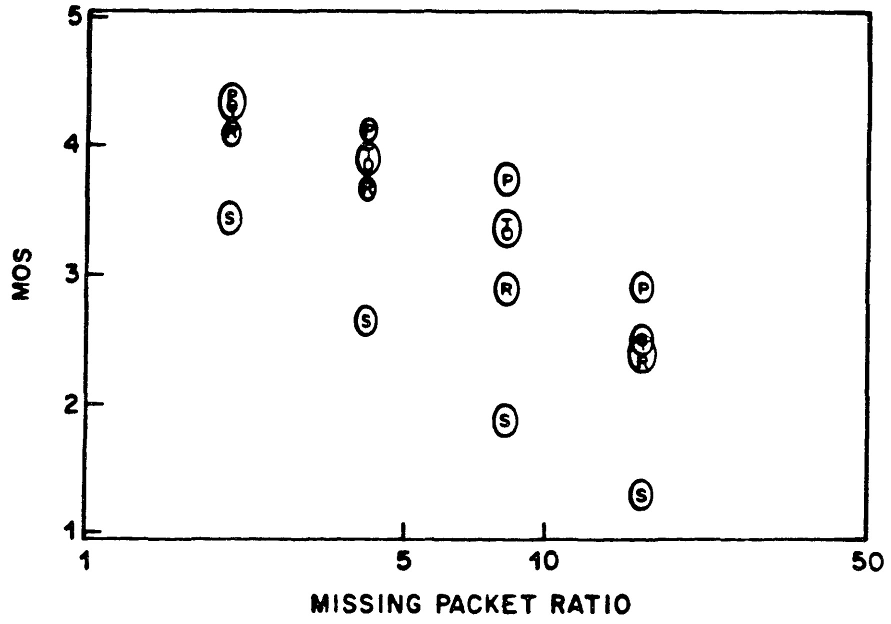


**Figure 24.** MOS vs. missing packet ratio for each process: see Table 5 for key to symbols. Processes sharing an ellipse are not significantly different.



from that of another process with which it does not share an ellipse. Overall, pitch detection shows up best, then pattern matching, followed by repeating the previous packet, and the worst is substituting silence. As an engineer, I'd say that if the implementation allows for the increased computation, pattern matching is better to use than repeating the previous packet, because of the significant improvement in quality that results. Whether to use pitch detection or pattern matching also depends upon the implementation. If the implementation is such that computation and delay are about the same for the two methods, pitch detection would certainly be better, as it produces significantly better quality.

### 4.2 Results for the ADPCM Algorithms

Again, I made several hypotheses before analyzing the data. I expected that it would be possible to combine both versions of the test again. Unfortunately, this did not prove true, and I had to analyze the morning and afternoon sessions separately, as shown in Appendix B. Also, I reasoned that the best way to pattern-match the ADPCM speech would be to pattern-match the decoder parameters, too. That is, after selecting a segment of speech to plug into a gap, one would want to restore the decoder parameters to what they had been at the end of that segment. Since that segment of speech is merged into the next packet so that the amplitude and phase vary smoothly, the decoder would be ready for speech to continue from the amplitude and phase of the end of the reconstruction segment. At present it would be very difficult to implement a practical decoder capable of recording 32 ms of the past history of the quantizer step size and predictor coefficients. However, based on this reasoning, I expected repeating the previous packet coupled with freezing the parameters, and pattern matching coupled with updating the parameters, to be significantly better than the other algorithms. Based on listening experience, I expected substituting silence to be significantly worse than anything that reset decoder parameters, which in turn would be significantly worse than the remaining algorithms.

Table 8 shows, for each missing packet ratio in the morning test, which processes had MOS's which were significantly higher than those for other processes. Table 7 assigns a two-letter name to each process for the purpose of making Table 8 concise. Table 9 shows the analysis results for the afternoon session.

TABLE 7

| Name | Process |
|---|---|
| PF | pattern matching, freezing decoder parameters |
| PR | pattern matching, resetting decoder parameters |
| PU | pattern matching, updating decoder parameters |
| RF | repeating previous packet, freezing decoder parameters |
| RR | repeating previous packet, resetting decoder parameters |
| RU | repeating previous packet, updating decoder parameters |
| SF | substituting silence, freezing decoder parameters |

**TABLE 8.** morning session

| Missing Packet Rate | On each line, processes on left are significantly better than processes on right. | |
|---|---|---|
| 1% | PR,PF,RF,PU,RU,RR | SF |
| 2% | PU,PF | RR,PR,SF |
| | RU,RF,RR | PR,SF |
| | PR | SF |
| 4% | PU,RU,PF,RF | PR,RR,SF |
| | PR,RR | SF |
| 8% | PF | RF,RR,PR,SF |
| | RU,PR,RF | RR,PR,SF |
| | RR,PR | SF |

**TABLE 9.** afternoon session

| Missing Packet Rate | On each line, processes on left are significantly better than processes on right. | |
|---|---|---|
| 1% | RU | PR,RR,SF |
| | RF | RR,SF |
| | PU,PF | SF |
| 2% | PF,PU,RF,RU | PR,SF,RR |
| 4% | RU,PU,RF,PF | RR,PR,SF |
| | RR | SF |
| 8% | RU,PU,RF,PF | RR,PR,SF |
| | RR | SF |

The results indicate that the best quality comes from repeating the previous packet or pattern matching, coupled with freezing or updating the decoder parameters. Obviously, for ADPCM speech, my pattern matching algorithm is not as good as the simpler method of repeating the previous packet, because it requires more computation to produce the same quality. Similarly, freezing the parameters is better than updating them, because it produces the same quality for less computation. Therefore, repeating the previous packet and freezing the decoder parameters is the best thing to do, since it involves the least computation. Figures 25-30 show plots of the MOS scores of various processes at various missing packet ratios.

**Figure 25.** MOS vs. *SNR* for reference noise process for the morning session

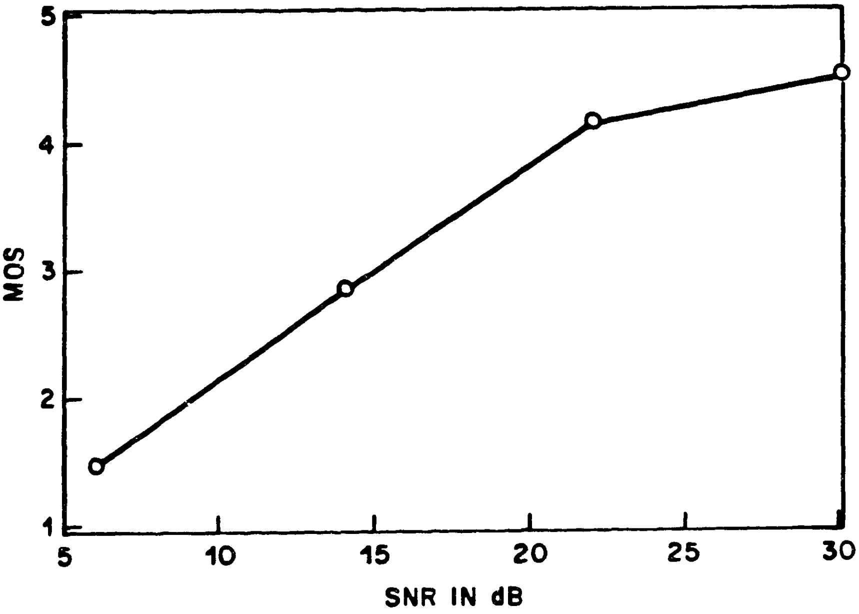
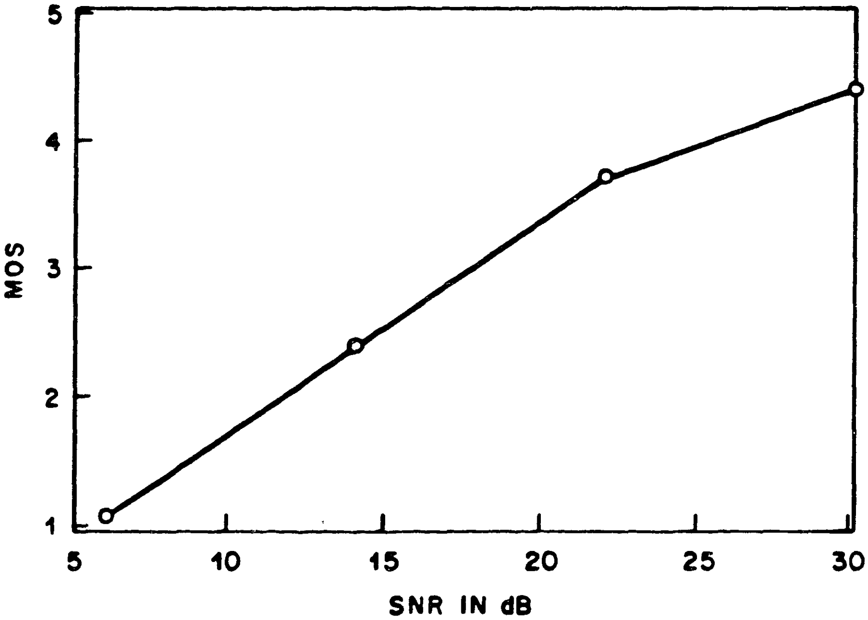


**Figure 26.** MOS vs. *SNR* for reference noise process for the afternoon session



Figures 25 and 26 show the MOS's for the reference noise process at each SNR, for each version of the test. Figures 27 and 28 show MOS vs. missing packet ratio for each process and each

**Figure 27.** MOS vs. missing packet ratio for each process for the morning session: see Table 7 for key to symbols

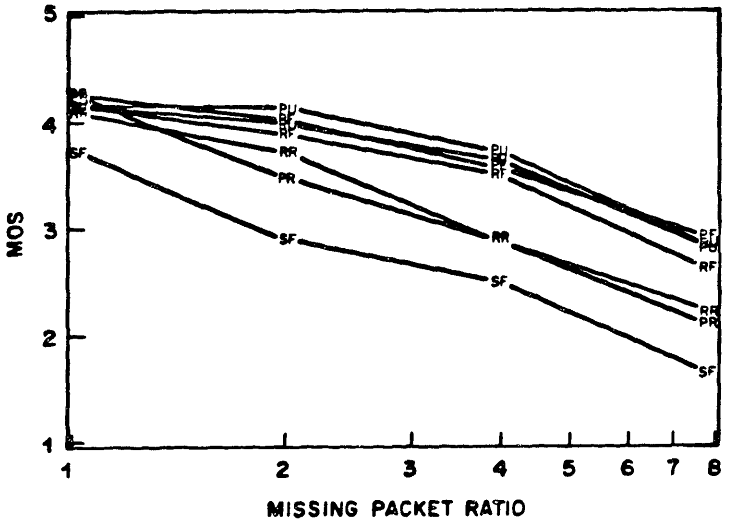
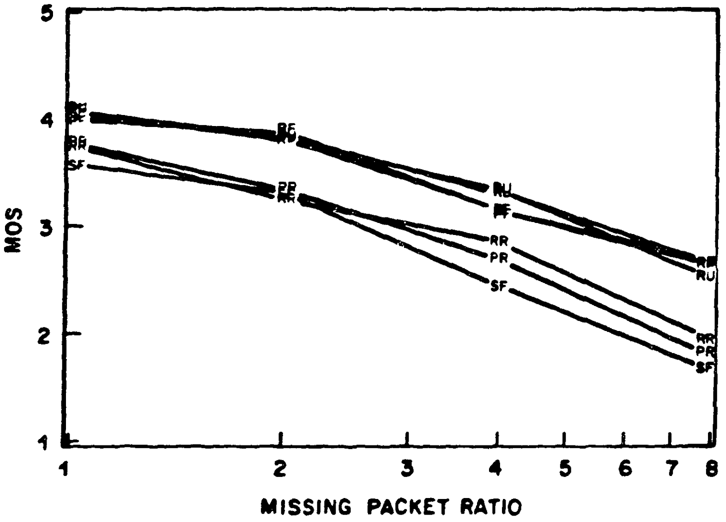


**Figure 28.** MOS vs. missing packet ratio for each process for the afternoon session: see Table 7 for key to symbols



version of the test, and Figures 29 and 30 show scatter plots of Figures 27 and 28 respectively, but with ellipses drawn on them. Each process' MOS is significantly different from that of another process with which it does not share an ellipse. These figures clearly illustrate how the processes break up into: 1) combinations of pattern matching or repeating the previous packet, and freezing or updating the decoder parameters; 2) resetting the decoder pa..ameters; and 3) substituting silence.

**Figure 29.** MOS vs. missing packet ratio for each process for the morning session: see Table 7 for key to symbols. Processes sharing an ellipse are not significantly different.

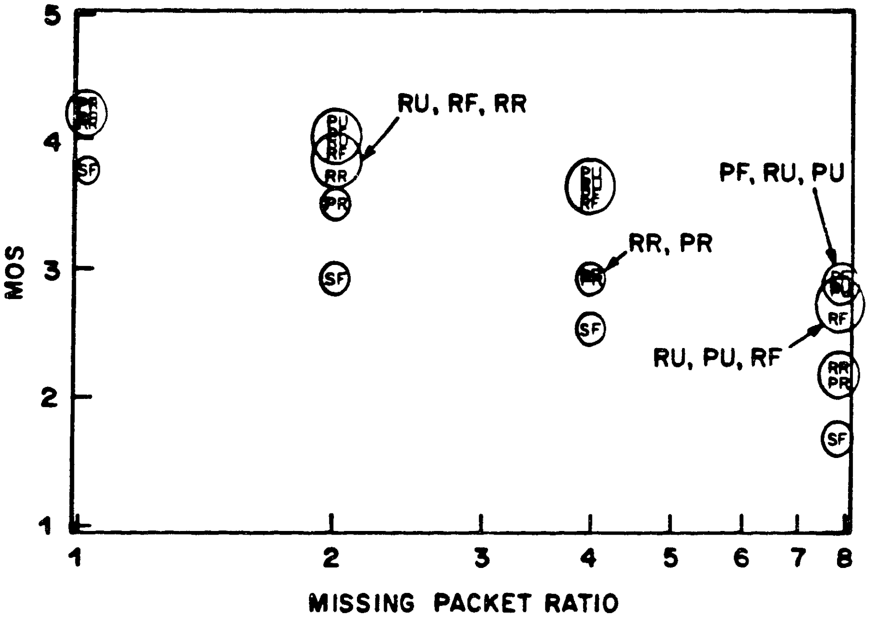
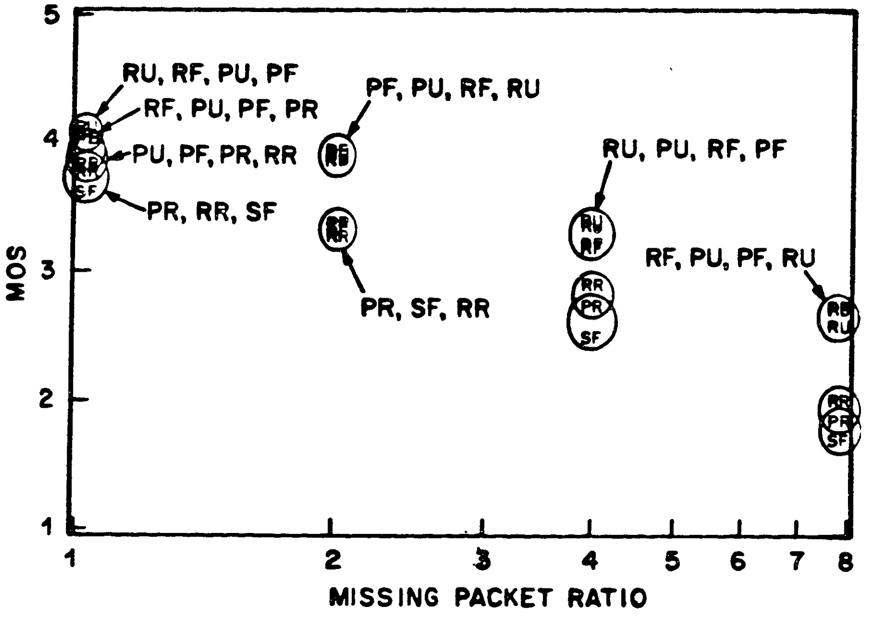


**Figure 30.** MOS vs. missing packet ratio for each process for the afternoon session: see Table 7 for key to symbols. Processes sharing an ellipse are not significantly different.



## 5. Conclusion

In this thesis I developed and tested the quality of various algorithms for reconstructing missing packets of PCM and ADPCM speech. These algorithms may prove useful in several ways. They can be used to reconstruct packets in existing networks where packets are dropped due to excessive delay caused by network congestion, or where packets or headers are garbled due to noise or fading in the channel. That is, they help to recover from accidental packet loss. Moreover, they can also be used to increase the effective throughput of a network by dropping packets on purpose and reconstructing them at the other end. Since the algorithms work best when only isolated packets are dropped, a protocol could be developed which would spread the dropped packets uniformly in time and among network users.

For PCM speech, replicating the last pitch period of the packet preceding the missing one for the length of the missing packet produces the best quality of speech. One-sided pattern matching as described in Section 2.1 never does worse than repeating the previous packet, and sometimes it does better. All three of these methods are better than filling missing packets with silence. Surprisingly, merging the reconstructed packet with the surrounding speech did not significantly improve the quality of the speech.

For ADPCM speech, reconstruction should definitely be done on the decoder's PCM output, rather than its ADPCM input. Reconstruction on the input can produce loud pops which can be painful to the listener. Filling missing packets with silence or resetting the decoder parameters produces the worst quality speech. The methods which work the best are combinations of repeating the previous packet or pattern matching, and freezing or updating the decoder parameters. I recommend repeating the previous packet and freezing the decoder parameters, since it requires less computation and delay.

I have five suggestions for future research:

(1) Better methods of merging should be explored. Things such as different weighting envelopes or filters might work. I am sure that a good part of the distortion comes from the edges of the reconstruction packet, and that better merging could reduce this distortion.

(2) Pitch detection could be implemented in an ADPCM algorithm, to find out if it would improve things further.

(3) Speaker dependences among the various PCM and ADPCM processes should be explored. In a real network, one would not want to use algorithms that are noticeably worse for particular types of voices, as this would be disturbing to the user.

(4) The implementation of a codec providing storage of a long sequence of codec parameters should be explored in order to find out whether pattern matching in conjunction with restoring old values of decoder parameters would indeed yield significant improvement.

(5) Research could be done on which types of packets (voiced or unvoiced, high or low energy, etc.) can be most easily reconstructed. Then a network protocol could be developed which would drop the packets which are most easily recovered, and then reconstruct them at the other end.

## *APPENDIX A*

### *Test Instructions*

### *Crawford Hill Test - HOH (11-85)*

The experiment in which you are about to participate is designed to examine the effects of a variety of telephone transmission impairments on audio quality. We will play recordings of people speaking sets of three English sentences. Your task is to listen to the test conditions as if they were telephone connections that you might actually use for normal purposes.

Immediately after each test condition, you are to make a judgement of the quality of the speech you just heard. Judgements are to be made in one of five categories, as labeled in front of the buttons at your station: **Excellent, Good, Fair, Poor or Unsatisfactory.**

The experiment will proceed as follows: Please look at the keyboard in front of you. When the yellow light comes on, you are to listen for a sample of speech. Immediately after the speech sample, the yellow light will go out. Very quickly, the green light will come on. At this time you are to rate the quality of the telephone connection you heard by pressing one of the buttons labeled Excellent through Unsatisfactory. Please remember, you must wait for the green light to come on before you press any button. The green light will only be on for a few seconds, so please rate the quality by pressing one of the buttons promptly so that your vote is not lost. Please hold down the button until the green light goes out. After you make a rating, the red light will come on, there will be short pause and then you will hear another sentence set, and so on.

Please try to be both attentive and consistent in your judgements during the experiment.

Let's try a short practice session of 12 sentence sets to familiarize you with the procedure. The test conditions you are about to hear will also expose you to the range of telephone quality you will experience during the real data collection. The practice sentences will include examples of the best and the worst audio quality you will hear in this experiment.

Remember, the basic task is to rate the quality of the speech samples as if they were being received over ordinary telephone connections.

Any questions?

# *APPENDIX B*

## *Data Analysis*

Since some of the subjects had missed entering their responses to certain conditions, I replaced all of the null responses with the averages of the other responses given for the same condition in the same version of the test. This made the data easier to manipulate, without changing the means at all. It also had negligible effect on the variances for my purpose.

### *Analyzing Results for the PCM Algorithms*

In the subjective test for the PCM algorithms, there were ten null responses in the morning session, and none in the afternoon session. There were some software problems getting the first session started, so that those subjects were at the test site twenty minutes longer than the others. This may be part of the cause of the late responses. Before analyzing the data, I hypothesized that it would be possible to combine both versions of the test as samples taken from the same population. This hypothesis was based on the premise that the variability due to permutation order, missing packet order, and other differences between the two sessions would show no systematic patterns and could therefore be considered as part of the random variability inherent in subjective tests. Furthermore, in order to simplify the analysis, I assumed that the speaker effects would be the same for every process.

Effects due to the pattern of dropped packets and the permutation of the conditions can be averaged out to some extent by lumping the two versions of the test as samples taken from a single population with a single probability distribution. Contributors to the variance of the responses in the two versions include: ten different processes, four different missing packet ratios (or SNR's), eight different speakers, ten or eleven different listeners, 320 different random seeds, the permutation of the conditions, and environmental factors such as the time of day, temperature of the room, length of the snack break, interruptions of the test (there was one interruption in the morning session), and the like. Of these factors, the ones which were common to both versions of the test were the ten processes, four missing packet ratios (SNR's), and eight speakers. Therefore, I cannot assume the variances were the same in both versions.

Let $\sigma_{am}^2$ and $\sigma_{pm}^2$ be the true variances of the probability distributions of all of the responses in the morning session and the afternoon session respectively, and let $s_{am}^2$ and $s_{pm}^2$ be their respective sample variances. An F test of sample variances tests the hypothesis that $\sigma_{am}^2 = \sigma_{pm}^2$. [10] I choose a significance level $\alpha=0.01$ so that if the probability that $\sigma_{am}^2 = \sigma_{pm}^2$ is less than 0.01, I will reject my hypothesis.

For my data, $s_{am}^2 = 1.37$, and $s_{pm}^2 = 1.41$. There were 3520 responses given in the morning, and 3200 in the afternoon. Therefore, $F_s = s_{pm}^2 / s_{am}^2$, with 3199 degrees of freedom for the numerator, and 3519 degrees of freedom for the denominator. The resultant probability for $F$ is $p_F = Pr[F \geq F_s] + Pr[F \leq 1/F_s] = 0.441$. Therefore, I can assume the variances of the two versions are the same.

Given this assumption, Student's t-test will test the hypothesis $\mu_{am} = \mu_{pm}$ where $\mu_{am}$ and $\mu_{pm}$ are the true means of the probability distributions of all of the responses in the morning and afternoon sessions respectively. [10] I test this hypothesis with a significance level $\alpha=0.01$. Let $m_{am}$ and $m_{pm}$ be the sample means, or mean opinion scores (MOS's), in the morning and afternoon respectively, and let $d$ be the difference between them. From the data, $m_{am} = 3.29$, $m_{pm} = 3.23$, and $d = 0.06$. The "pooled variance" is

$$s^2 = \frac{3519 s_{am}^2 + 3199 s_{pm}^2}{6718} = 1.39 \ , \tag{B-1}$$

and the variance for $d$ is

$$s_d^2 = \frac{s^2}{3520} + \frac{s^2}{3200} = 8.30 \times 10^{-4} \ . \tag{B-2}$$

This results in

$$t_d = \frac{m_{am} - m_{pm}}{s_d} = 2.09 \tag{B-3}$$

with probability $p_t = Pr[t \geq t_d] = 0.037$.

Thus, the difference between the means of the two versions of the test is insignificant, and the variances of the two versions can be assumed equal. Therefore, the results of the two tests can be pooled.

An F-test of sample variances performed on the variances of every pair of processes for each missing packet ratio showed that all of the variances could not be assumed equal. This can be expected, as at some ratios some processes were so obviously good or bad that nearly everyone's responses agreed, while the responses for other processes varied more. Therefore, this discrepancy in variances does not necessarily disprove the assumption that the speaker effects are the same for every process. At each missing packet ratio, the variance and MOS of each process involved 168 samples (21 listeners × 8 speakers), or 167 degrees of freedom, and the significance level was 0.01.

For pairs of processes whose variances could be assumed equal, I performed a t-test on the difference between their MOS's, using equations (B-1)-(B-3) to discover if the difference is significant. For pairs of processes whose variances could not be assumed equal, I used an approximation to Student's t-test to find out whether the difference between their MOS's is significant [10]. The approximation to t is

$$t = \frac{m_1 - m_2}{s_1^2/n_1 + s_2^2/n_2} \tag{B-4}$$

where $m_x$ is the sample mean of process $x$, $s_x$ is the sample variance of process $x$, and $n_x$ is the number of samples contributing to the MOS of process $x$. Instead of using $n_x - 1$ as the degrees of freedom, when the variances of the two processes cannot be assumed equal I use

$$df = \frac{[(s_1^2/n_1) + (s_2^2/n_2)]^2}{[(s_1^2/n_1)^2/(n_1 - 1)] + [(s_2^2/n_2)^2/(n_2 - 1)]} \tag{B-5}$$

where $df$ stands for degrees of freedom.

*Analyzing Results for the ADPCM Algorithms*

In the subjective test for the ADPCM algorithms, there were four null responses in the morning session, and none in the afternoon session.

I expected that it would be possible to combine both versions of the test again. Furthermore, in order to simplify the analysis, I again assumed that the speaker effects would be the same for every process.

Things which contribute to the variance of the responses in the two versions of this test are: eight different processes, four different missing packet ratios (or SNR's), eight different speakers, eleven different listeners, 256 different random seeds, the permutation of the conditions, and environmental factors. Of these factors, the ones which were common to both versions of the test were again the eight processes, four missing packet ratios (SNR's), and eight speakers. Therefore, I cannot assume the variances were the same in both versions.

Let $\sigma_{am}^2$ and $\sigma_{pm}^2$ be the true variances of the probability distributions of all of the responses in the morning session and the afternoon session respectively. An F test of sample variances tests the hypothesis that $\sigma_{am}^2 = \sigma_{pm}^2$. [10] For this test it turns out that the two versions cannot be pooled. The variances can be assumed equal, but according to Student's t-test, the means cannot be assumed equal. The means were 3.36 for the morning session, and 3.16 for the afternoon session. Again, I performed an F-test of sample variances on the variances of every pair of processes for each missing packet ratio. However, this time I treated each version of the test separately. As before, all of the variances could not be assumed equal. At each ratio, for each version of the test, the variance and MOS of each process involved 88 samples (11 listeners × 8 speakers), or 87 degrees of freedom, and the significance level was 0.01.

For pairs of processes whose variances could be assumed equal, I performed a t-test on the difference between their MOS's, using equations (B-1)-(B-3) to discover if the difference is significant. For pairs of processes whose variances could not be assumed equal, I used an approximation to Student's t-test, as described above in the PCM analysis, to find out whether the difference between their MOS's is significant [10].

## *REFERENCES*

1. Thomas E. Stern, "Analysis of Packet Voice on Local Area Networks," IEEE International Conference on Communications 1985, pp.272-277

2. C.J. Weinstein and J. W. Forgie, "Experience with Speech Communication in Packet Networks," IEEE Journal on Selected Areas in Communications, Vol. SAC-1 No. 6, December 1983, pp. 963-980

3. R. F. Rous and P. J. See, "Performance Considerations and Protocols for Packeted Speech," IERE Conference on Digital Processing of Signals in Communications No. 62, Loughborough University, April 1985, pp. 221-226

4. Nuggehally S. Jayant, Susan W. Christensen, "Effects of Packet Losses in Waveform Coded Speech and Improvements Due to an Odd-Even Sample-Interpolation Procedure," IEEE Transactions on Communications, Vol. COM-29, No. 2, February 1981, pp. 101-109

5. R. C. F. Tucker and J. E. Flood, "Optimizing the Performance of Packet-Switched Speech," IERE Conference on Digital Processing of Signals in Communications No. 62, Loughborough University, April 1985, pp. 227-234

6. A. W. F. Huggins, "Effect of Lost Packets on Speech Intelligibility," NSC Note 78, 1976

7. D. J. Goodman, O. G. Jaffe, G. B. Lockhart, and W. C. Wong, "Waveform Substitution Techniques for Recovering Missing Speech Segments in Packet Voice Communications," IEEE International Conference on Acoustics, Speech and Signal Processing 1986

8. Ad hoc group on 32 kbit/s ADPCM, Recommendation G.7zz: "32 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)," CCITT Study Group XVIII, Geneva, November, 1983

9. Richard A. Becker and John M. Chambers, *S: An Interactive Environment for Data Analysis and Graphics* , The Wadsworth Statistics/Probability Series, Bell Telephone Laboratories, 1984

10. Norman H. Nie, C. Hadlai Hull, Jean G. Jenkins, Karin Steinbrenner, Dale H. Bent, *Statistical Package for the Social Sciences* , second edition, McGraw Hill, 1975