Performance of Error Correcting Schemes for Broadcast Telidon

by

Dr. Brian Mortimer and

Dr. Michael Moore

Department of Mathematics and Statistics

Carleton University
Ottawa, Canada

Performance of Error Correcting Schemes for Broadcast Telidon

by

Dr. Brian Mortimer and
Dr. Michael Moore

Final Report
prepared under contract
DSS OST84 - 00172

for the

Department of Communications
Ottawa, Ontario, Canada.

March 29, 1985

Performance of Error Correcting Schemes for Broadcast Telidon

DSS Contract No. OST84-00172

Principal Investigator: Dr. Brian Mortimer
Dept. of Mathematics and Statistics
Carleton University
Ottawa Canada

Outside Consultant: Dr. Michael J. Moore,
2 Aurora Cresc.,
Nepean, Ontario.

Scientific Authority: Dr. Michael Sablatsh,
Communications Research Centre,
Dept. of Communications,
Ottawa, Ontario, Canada.

Delivered: (i) Progress Report — November, 1984
(ii) Final Report — March, 1985
(iii) Statistical Results on Channel Error Distributions and Code Performance — March, 1985.

Table of Contents

Chapter 1. Introduction and Results

1.1 Analysis of Field Data 1.1

1.2 Results and Conclusions on Code Performance 1.2

1.3 Bundle Lengths 1.4

1.4 Error Distributions in the Channel 1.5

Chapter 2. Comparing the Performance of Various Codes

2.1 The Codes 2.1

2.2 The Grading SYStem 2.3

2.3 Results 2.4

Chapter 3. Decoding Strategies for the Bundle Code

3.1 Decoding Strategies 3.1

3.2 Bundle Length Effects 3.4

3.3 Results of Stategies: Recommendations 3.19

Chapter 4. The Distribution of Errors in the Channel

4.1 The Channel 4.1

4.2 Bit Error Rate 4.5

4.3 Smeared Packets 4.12

4.4 The Bytes Within a Packet 4.16

4.5 Bits in a Byte 4.23

4.6 Effects Between Packets 4.26

References

Chapter 1. Introduction and Results

1.1 Analysis of Field Data.

In any system of broadcast teletext, some corruptions of the data transmitted is almost inevitable. In previous reports [1]-[5], we have made theoretical studies of various error detection and correction schemes and their performances when confronted with randomly distributed errors. The reasons for this choice of error pattern are that most other schemes have been measured against it and that it is fairly amendable to computation. Other models have been examined [5] but suffer from the disadvantage that nobody knows if they approximate what occurs during real transmissions. Because of this, all conclusions have, of necessity, had to be of a very tentative nature.

In this report we can now concentrate on data which has been obtained from actual channels of transmission. Our analysis has taken two main thrusts. Firstly we have analyzed the performance of a variety of codes in their attempts to detect and correct errors which occur in the channels. The codes are detailed in Chapter 2 and consist of codes on individual packets with various capabilities plus two other coding options called "bundle codes" where certain complete packets are utilized for error protection.

For the coding of individual packets, whether or not the code will be successful can be determined if the number and distribution of the errors within the packet is known. Hence the analysis for these codes was obtained from the determination of the error distributions within the channel. We therefore wrote software to process the site data from the various sites and produce the site data from the various sites and produce the error statistics needed. Further processing then enables the performance of each code to be evaluated.

When packets are bundled together to provide extra protection the situation is more difficult to assess. In our last report [1] , in the second chapter we detail how the precise location of a small number of errors can affect the outcome of the decoding attempt. To complete our analysis of the behavious of the bundle codes we therefore used actual decoders and proceeded to decode data which had been corrputed with the error patterns obtained at the various receiving sites. The overall results are examined in Chapter 3 while the results at individual sites are in Chapter 2, table 2.2.

The second stage of our analysis was to determine the error distributions and to look for correlations. Sometimes a casual investigation of the data appeared to indicate that a particular bit of every byte was favoured as an error location within a packet. For example, many errors in one packet might appear in bit 8 of the corrupted bytes. We therefore tested to see if a particular bit was indeed favoured over the others in the byte. We also tested to see if particular bytes showed a tendency to contain errors. The results of this investigation can be found in Chapter 4.

1.2 Results and Conclusions on Code Performance

One of the first statistics to calculate for any particular channel is the Bit Error Rate (B.E.R.), as this gives a good guide as to how the codes will perform. We tested to see how the B.E.R. varied with time at each site and this lead to the detection of certain anomalous packets where the rate was close to 50%. Since the most likely explanation for this phenomenon is that the teletext decoder lost synchronization we refer to these packets, where at least 20 bytes are in error, as "smeared" packets. The performance of the coding schemes is largely determined by the B.E.R. and the number of smeared packets.

When there are smeared packets or rejected packets, only the bundle codes can re-insert these packets and so are the only codes which have any possibility of accpeting a page where these packets are present. When there are too many smeared packets or they occur too close together then nothing can cope with them. The strategy of decoding a bundle code by replacing one or two smeared packets with erasures proved effective at correcting isolated smeared packets.

When assessing the performances of the various codes, we used the rate of rejection of a page of data as the criterion for ranking them. The number of packets which form a page varies from code to code and each number was selected to provide a roughly constant amount of data to the page. We fairly arbitrarily selected a 1% rejection rate on the pages as the cut off for acceptability of a code, as this corresponds to the rate picked by the Japanese as an allowable bound [6+7].

We have ranked the codes at each of 35 of the sites for which we have data. At the remaining sites there were very few errors and all the coding schemes handled virtually 100% of the packets. When comparing the behaviour of the codes, the B.E.R. has been adjusted to remove the affects of the smeared packets to give a truer indication of the background rate.

Using the 1% bound for the page rejection rate, we found that all of the coding schemes fail when the adjusted B.E.R. exceeds 10^{-3} . At the sites in this range only the bundle codes and the Japanese codes could correctly decode 90% or more of the pages and even this was often unattainable. The double bundle code with repeated decodings survived at the largest number of sites.

When the B.E.R. is between 10^{-3} and 10^{-4} only the bundle codes and the Japanese code prove to be effective and as the rate falls towards 10^{-5} the less ambitious coding schemes become effective. When the B.E.R. is below 10^{-5}, then parity checking alone would be an acceptable strategy. In all conditions the double bundle code is always at least as good as the Japanese BEST code and sometimes outperforms it.

1.3 Bundle Lengths

The overhead for error protection in the bundle code can be reduced by increasing the number of packets of data sent for each packet of error protection. At a double bundle length of 14 there are 12 packets of data and we get an information rate of 62% (or 79% if we ignore the 8 byte packet headers). However, the rejection rate for the bundles also increases as the bundle length is increased. The bundle decoders were tested at various lengths and the rates of decoding errors and decoding failures were observed. A part from the rapid changes in the information rate at very small bundle lengths, it was found that the rejection rate increases more rapidly than the information rate. This offset means that there is no gain in increasing the bundle length beyond the range of 10 to 20.

1.4 Error Distributions in the Channel.

The second major part of the research project was devoted to studying the distribution of errors in the broadcast teletext channel. In particular we want to know if there is any structure to the occurrence of errors. If it turned out that errors arise as independent events then our theoretical predictions of code performance could have some validity. Unfortunately this is not what was actually observed.

Table 1.1 lists the sites used in our analysis. For each we have a Site Number and Location, the distance to the transmitter, the data and time of the test start, which of two receives (Rhodes and Schwartz or Sony) was used, the sample size in thousands of packets, the adjusted bit error rate, the smeared packet rate and an indication whether the adjusted bit error rate was constant or varying during the test. Note that Sites 373 and 374 were lost during processing so don't appear. The adjusted B.E.R. is the fraction of bits in error excluding bits in smeared packets and in an error free segment of 1000 packets. A smeared packet is one with at least 20 (and generally all 28) bytes in error.

At these sites the following were studied:

(i) the bit error rate vs. distance to the transmitter,
(ii) time variations in the bit error rate,
(iii) the rate at which smeared packets arise and their correlation to A.B.E.R.,
(iv) the distribution of erroneous bytes in a packet,
(v) the distribution of erroneous bits in a byte,
(vi) the distribution of erroneous packets in the channel.

Table 1.1 Field Test Sites

Number	Location	Distance to Transmitter	Date-Time	Receiver or Mode	Sample Size x 10^3	Adjusted Bit Error Rate	Smeared Packet Rate	
Camp Fortune:			June, 1984					
300-1	Wakefield	13	19-10:32	Synch.	27.3	6.7e - 4	0.0	C
300-2	Wakefield	13	19-10:42	Zenith	27.6	4.3e - 4	0.0	C
301-1	N.D.de la Salette	33	19	Synch.	17.0	1.7e - 3	0.0	C
301-2	N.D.de la Salette	33	19-12:29	Zenith	18.3	2.6e - 3	0.0	C
302	Val des Bois	50	19	Synch.	11.0	5.1e - 3	0.0	C
Saskatoon:			November, 1984					
361	Leask	90	16-13:53	Sony	34.8	1.8e - 5	0.0	C
362	Parkside	110	16-15:02	R & S	23.1	1.4e - 4	3.5e-4	C
363	Wakaw	80	17-12:51	R & S	16.3	3.8e - 5	3.0e-4	V
364	Crystal Springs	110	17-15:11	R & S	43.7	1.2e - 5	1.3e-4	V
365	Birch Hills	120	17-16:59	R & S	14.2	1.0e - 4	9.8e-4	V
366	Colonsay	50	18-12:20	R & S	14.1	1.3e - 5	0.0	V
367	Watrous	90	18-14:56	R & S	13.9	4.5e - 5	6.7e-4	V
368	Watrous	90	18-15:10	Sony	11.6	1.1e - 5	0.0	V
369	Dundern	40	20-12:59	R & S	13.1	3.1e - 5	7.7e-4	V
370	Hanley	55	20-14:25	R & S	7.1	3.4e - 3	1.3e-2	C
371	Delisle	55	21-10:55	R & S	12.0	1.2e - 5	7.5e-4	V
372	Harris	85	21-12:55	R & S	28.1	8.6e - 5	6.4e-4	C
375	Asquith	50	21-19:01	R & S	20.0	3.8e - 5	2.2e-4	V
376	Radisson	60	23-11:20	R & S	16.9	2.0e - 5	0.0	
377	Maymont	85	23-12:34	R & S	29.3	6.6e - 5	2.3e-4	
378	Maymont	85	23-13:42	R & S	20.0	5.6e - 6	2.5e-4	V

North Battleford:								
379	North Battleford (L.O.S.)	3	24-10:35	R & S	no synchronization			
380	North Battleford (L.O.S.)	3	24-11:00	Sony	11.3	3.7e - 3	8.9e-2	C
381	North Battleford (L.O.S.)	3	24-11:53	R & S	0.34	3.3e - 3	8.6e-2	C
382	Highway 29	15	24-15:45	Sony	17.5	1.5e - 3	1.1e-4	C
383	unknown		24-16:18	R & S	1.7	5.8e - 3	1.7e-2	C
Stranraer:								
384	Stranraer (L.O.S.)	3	25-12:32	R & S	40.4	8.0e - 6	0.0	C
385	Stranraer (L.O.S)	3	25-14:16	Sony	16.6	3.1e - 5	0.0	V
386	Kerrobert	50	25-15:31	R & S	2.3	5.1e - 3	2.8e-2	C
387	Netherhill	30	25-18:07	R & S	19.7	8.6e - 4	3.0e-3	V
388	Rosetown	20	26-10:33	R & S	25.4	3.5e - 4	2.0e-3	C
389	Hershel	10	26-11:37	R & S	42.5	2.1e - 4	0.0	C
390	Hershel	10	26-12:07	Sony	18.4	2.8e - 5	0.0	V
391	Plenty	10	26-14:18	R & S	43.6	1.7e - 5	0.0	C
392	Druid	23	26-15:40	R & S	22.9	1.6e - 5	0.0	C
393	Druid	23	26-16:09	R & S	35.6	1.0e - 5	0.0	C

We have made the following observations.

(i) At high bit error rates there is an increase in bit error rate with increasing distance to the transmitter. At low BER ($\leq 10^4$) there is no relationship between BER and distance to the transmitter site.

(ii) There are many sites where the BER varies significantly over time, variations lasting on the order of a few minutes on the channel.

(iii) There were smeared packets at many sites particularly but not always at sites with high BER. There are some very noisy sites with no smeared packets. At a majority of sites there is a strong correlation between the smeared packet rate (SPR) and the A.B.E.R., the former being on the order of 10 times the later. With a small variance the bit error rate inside smeared packets was 0.467. The smeared packets appear to be produced by the same stochastic process as the random error.

(iv) At 8 Sites (25%) there was a definite increase in the likelihood of an error as we proceed down the packet in time. Of these sites. 5 consist of all the sites from the Camp Fortune transmitter.

(v) At a remarkable number of sites (50%) bit 7 is more likely, by at least 60%, to be in error than any other bit. Here we are numbering from 1 to 8.

(vi) Packets are very widely separated in the channel. We should expect the erroneous packets to arise independently at most sites. This has been an interesting and difficult question to answer. At all but 4 very noisy sites the gaps between erroneous packets formed a random time series. On the other hand a significant number of sites showed a definite deviation from the expected distribution of gap lengths. So erroneous packets are not arising exactly independently in the channel and the nature of the distribution has not yet been unravelled.

Details on all these points are given in Chapter 4.

Chapter 2. Comparing the Performance of Various Codes.

2.1 The Codes

We have measured the performance of a number of different error correcting codes on the various sets of field data. In fact we are assessing decoding strategies not particular codes. Thus we have not calculated rates of decoding errors for individual codes.

The following are the codes:

(i) No Code: only parity checks on bytes are used to detect errors.

(ii) $\leq$ 1 bit error : single bit error correcting code, for example PRODUCT code.

(iii) $\leq$ 2 bit error : double bit error correcting code; for example Code C could be decoded this way.

(iv) $\leq$ 4 bit errors : all patterns of at most four bit errors are corrected. This case is used to provide an intermediate case to (v). The code could be some sort of BCH code for example.

(v) $\leq$ 8 bit errors : all patterns of at most eight bit errors are corrected. This is the case of the Japanese code BESI. This is a majority-logic decodable shortened cyclic code with 82 check digits.

(vi) 1 byte : all errors confined to one byte are corrected. Code C could be decoded this way.

(vii) 1 byte - 2 erasures: either a single byte or two bytes each with a parity failure are corrected. This is the sort of decoding used by Code C in the Bundle code.

(viii) Bunde I-2 : the Single Bundle code with two pass decoding. Thus the data bytes are decoded twice: horizontally, vertically, horizontally, vertically. Decoding failures (horizontally) are not written in as erasures.

(ix) Bunde II-2 : the Double Bundle code with two pass decoding. The same remarks as for (viii).

For the test we break up the packets into pages of a size dependent on the code being considered. The size is chosen to keep a more or less constant amount of information on the page. The standard is set to agree with the Japanese choice so that straight forward comparisions with their work can be made.

Table 2.1 Page Lengths Assigned to the Codes.

(i) No Code	10
(ii) $\leq$ 1 bit	10
(iii) $\leq$ 2 bits	10
(iv) $\leq$ 4 bits	11
(v) $\leq$ 8 bits	12
(vi) 1 byte	10
(vii) 1 byte - 2 erasures	10
(viii) Bundle I-2	14
(ix) Bundle II-2	14

The assignment is not equally fair to all codes since we have to use complete packets. It is intentionally hardest on the Bundle codes making their pages a couple of packets longer than strictly necessary. The idea behind the Japanese page length is to convey a single screen of text information on one page.

The pages are taken successively in the channel. Thus the first page is the first h packets, say, and the second page is packets h+1 through 2h . We could have obtained h different sets of pages from the same data by leaving out the first i packets (for i=0, 1,..., h-1) and then taking successive runs of h packets as pages. We could then take the average performance as a statistic with less variance. We haven't tried this technique on the data as yet, but it would remove some, at least, of the rough edges (see for example Site 367).

2.2 The Grading System.

At each site the probability of a correctly decoded page was calculated for each of the ten codes. Since there is a certain amount of variance in this number and different sites are often quite different it was decided to use a discrete scale or grading system. If X is the observed frequency of pages correctly decoded then the grades are assigned as follows:

$X = 100\%$	A+
$99.9\% \leq X < 100\%$	A
$99.5\% \leq X < 99.5\%$	B
$99\% \leq X < 99.5\%$	C
$95\% \leq X < 99\%$	D
$90\% \leq X < 95\%$	F
$X < 90\%$	FNS*

---

* "FNS" stands for "failure - no supplemental"; loosely speaking, failure without hope of redemption.

The system was chosen more or less arbitrarily except that the Japanese criterion neatly translates as "grades of C or better are acceptable" (Yamada [6]),
See also [7]

2.3 Results

The results for 35 sites are collected in Table 2.2. The sites are listed in a rough ranking from best to worst. The codes are ranked from worst to best from left to right. The (adjusted) bit error rate at each site is included.

There is a correlation between degrading performance and increasing bit error rates. We see that at about ABER = 10^{-3} and up most of the codes fall apart with the Double Bundle code continuing to work well to about ABER = 3×10^{-3} . The difficulties at these bit error rates include both large numbers of random errors and numerous smeared packets.

There appears to be another division point at about ABER = 10^{-4} . This is about where the single packet codes become unacceptable. ie. the Product Code or Code C decoded various ways. Below ABER = 10^{-5} it appears that error detection is enough to give the required performance.

One may note that at some sites one code works very much better than another while at other sites the codes have more or less the same performance. Some of the factors involved here are the wide range of sample sizes and the clustering of smeared packets at some sites.

For example, Site 378 is a sample of 19,953 packets. There was a single bit error then 16000 error free packets followed by a smeared packet and 28 lost packets with 4 more random bit errors in the next 3000 packets. This creates four pages that no code can correct and all other pages error free.

Segment	Number of Bit Errors	B.E.R. on Segment	Number of Smeared Packets
1 :	10	4.4642857E-005	1 smeared packets
2 :	9	4.0178571E-005	
3 :	7	3.1250000E-005	
4 :	14	6.2500000E-005	
5 :	15	6.6964286E-005	
6 :	27	1.2053571E-004	3 smeared packets
7 :	12	5.3571429E-005	2 smeared packets
8 :	10	4.4642857E-005	
9 :	23	1.0267857E-004	1 smeared packets
10 :	25	1.1160714E-004	1 smeared packets
11 :	25	1.1160714E-004	
12 :	25	1.1160714E-004	1 smeared packets
13 :	16	7.1428571E-005	
14 :	10	4.4642857E-005	1 smeared packets
15 :	17	7.5892857E-005	2 smeared packets
16 :	28	1.2500000E-004	1 smeared packets
17 :	27	1.2053571E-004	
18 :	24	1.0714286E-004	1 smeared packets
19 :	34	1.5178571E-004	
20 :	15	6.6964286E-005	2 smeared packets
21 :	29	1.2946429E-004	
22 :	20	8.9285714E-005	
23 :	15	6.6964286E-005	
24 :	23	1.0267857E-004	1 smeared packets
25 :	18	8.0357143E-005	
26 :	19	8.4821429E-005	
27 :	19	8.4821429E-005	
28 :	25	1.1160714E-004	1 smeared packets
29**	1	8.4231806E-005	

****

total # of packets: 28054
number of correct packets:27515
number of erroneous packets: 538
number of lost packets : 1
number of smeared packets :18

Prob. of a correct packet : 9.8079E-001

Error rate in smeared packets : 4.7371031746E-001
Adjusted BER = 8.6252553E-005
Smeared Packet R. = 6.4164261E-004
SPR / ABER = 7.4391144E+000

Overall Bit Error Rate : 3.9019137E-004

Segment	Number of Bit Errors	B.E.R. on Segment	Number of Smeared Packets
1 :	1	4.4642857E-006	
2 :	0	0.0000000E+000	
3 :	0	0.0000000E+000	
4 :	0	0.0000000E+000	
5 :	0	0.0000000E+000	
6 :	0	0.0000000E+000	
7 :	0	0.0000000E+000	
8 :	0	0.0000000E+000	
9 :	0	0.0000000E+000	
10 :	0	0.0000000E+000	
11 :	0	0.0000000E+000	
12 :	0	0.0000000E+000	
13 :	0	0.0000000E+000	
14 :	0	0.0000000E+000	
15 :	0	0.0000000E+000	
16 :	0	0.0000000E+000	
17 :	1	4.4642857E-006	1 smeared packets
18 :	1	4.4642857E-006	
19 :	2	8.9285714E-006	
20**	0	0.0000000E+000	

****

total # of packets: 19953
number of correct packets:19919
number of erroneous packets: 6
number of lost packets : 28
number of smeared packets :1

Prob. of a correct packet : 9.9830E-001

Error rate in smeared packets : 5.8928571429E-001
Adjusted BER = 5.5803571E-006
Smeared Packet R. = 2.5000000E-004
SPR / ABER = 4.4800000E+001

Overall Bit Error Rate : 3.0652390E-005

Site 372 on the other hand is more uniformly noisy with 18 smeared packets and random bit errors at a rate of 9×10^{-5}. The smeared packets are just far enough apart for the Double Bundle to correct all the pages while the other codes stumble on roughly the same sets of pages. Using the criterion of at most 1% of pages rejected (grade of C or better) we have numbers and percentages of acceptable sites as in Table 2.2.

Table 2.2 Number of Sites with Acceptable Performance for Nine Codes

No Code	4	11%
≤ 1 bit	18	51%
1 byte	19	54%
≤ 2 bits	19	54%
1 byte & 2 erasures	19	54%
≤ 4 bits	22	63%
≤ 8 bits	24	69%
Bundle I-2	27	77%
Bundle II-2	29	83%

Site No.	No Code	≤ 1 bit	1 byte	1 byte + 2 erasures	≤ 2 bits	≤ 4 bits	≤ 8 bits	Bundle I-2	Bundle II-2	A.B.E.R.
384	C	A	A	A	A	A+	A+	A+	A+	8.0e-6
393	C	B	A	A+	A+	A+	A+	A+	A+	1.0e-5
364	B	A	A	A	A	A	A	A	A	1.2e-5
378	B	B	B	B	B	B	B	B	B	5.6e-6
377	D	C	C	C	C	C	B	B	B	6.6e-5
368	D	A	A	A+	A+	A+	A+	A+	A+	1.1e-5
366	D	A	A	A+	A+	A+	A+	A+	A+	1.3e-5
376	D	B	B	B	B	A	A+	A+	A+	2.0e-5
385	D	B	B	B	B	A+	A+	A+	A+	3.1e-5
390	D	B	B	A	A	A+	A+	A+	A+	2.8e-5
391	D	C	B	A	B	A	A+	A+	A+	1.7e-5
392	D	C	C	B	B	B	B	A+	A+	1.6e-5
363	D	C	C	B	B	B	B	B	A+	3.8e-5
369	D	C	C	C	C	C	C	C	A+	3.1e-5
371	D	C	C	C	C	C	C	C	A+	1.2e-5
361	D	B	B	A	A	B	B	A	A	1.8e-5
375	F	C	C	B	B	B	B	B	A+	3.8e-5
367	F	C	C	C	C	B	C	C	C	4.5e-5
372	FNS	D	C	C	C	C	C	C	A+	8.6e-5
389	FNS	F	F	D	D	C	B	A+	A+	2.1e-4
300-1	FNS	FNS	FNS	D	D	C	B	A+	A+	6.7e-4
300-2	FNS	FNS	FNS	D	D	B	A	A	A	4.3e-4
362	FNS	F	D	D	D	D	C	C	A	1.4e-4
382	FNS	FNS	FNS	FNS	FNS	D	B	B	B	1.5e-3
365	FNS	D	D	D	D	D	D	C	A	1.0e-4
387	FNS	FNS	FNS	FNS	FNS	F	D	C	B	8.6e-4
388	FNS	FNS	FNS	F	F	F	D	D	A	3.5e-4
301-1	FNS	FNS	FNS	FNS	FNS	FNS	D	D	B	1.7e-3
301-2	FNS	FNS	FNS	FNS	FNS	FNS	F	F	B	2.6e-3
380	FNS	FNS	FNS	FNS	FNS	FNS	D	F	D	3.7e-3
302	FNS	FNS	FNS	FNS	FNS	FNS	FNS	FNS	D	5.1e-3
370	FNS	FNS	FNS	FNS	FNS	FNS	FNS	FNS	F	3.4e-3
381	FNS	FNS	FNS	FNS	FNS	FNS	FNS	FNS	FNS	3.3e-3
383	FNS	FNS	FNS	FNS	FNS	FNS	FNS	FNS	FNS	5.8e-3
386	FNS	FNS	FNS	FNS	FNS	FNS	FNS	FNS	FNS	5.1e-3

Table 2.2. Performance of Nine Codes at 35 sites

Chapter 3. Decoding Strategies for the Bundle Code

3.1 Decoding Strategies

Each code word of the Carleton code is capable of correcting the erasure of two bytes from the word. The simplest method of ensuring this is to perform error corrections whenever exactly two bytes show parity failures. Since these two bytes are identifiable, the decoder can calculate the corrections required to convert the string of bytes into a code word of the Carleton Code. This feature means that the code can correct a wide range of errors but does increase the risk of a decoding failure since a combination of two odd parity errors and an even parity error in the same word will always lead to the decoder being deceived into thinking it has corrected the code word when, in fact, it has further corrupted the two bytes which showed a parity failure. However, the risk is still small and is outweighed by the ability of the decoder to replace erased bytes which is the basis of the bundle codes ability to restore a lost packet (or two lost packets in the case of the double bundle).

The most straightforward decoding technique for the bundle and double bundle codes is to first decode each packet using the methods mentioned above. We call this stage "horizontal" decoding since our usual picture of the packets is to stack them vertically. After each packet has been decoded the decoder then moves to "vertical" decoding where the code words now run across the various packets. In the case of the bundle code there are two bytes from each packet in every code word and in the double bundle there is only one byte.

If any of these vertical decodings proves to be unsuccessful, then we declare a failure of the bundle code concerned. Only if all of the codes succeed (or claim to succeed) is the bundle considered to have succeeded. Note that even though a horizontal codeword may be fooled into an erroneous decoding, this will often be corrected, or at least detected, by this vertical coding.

The decoding technique of horizontal, then vertical decoding we refer to as "1 pass decoding" and this technique was tested extensively at various bundle lengths to determine the effect of the lengths on the success rates for the codes. The second attack was to apply the decoder twice to every bundle. This we call "2 pass decoding" and it, too, was tried at many different bundle lengths. Between these two methods of decoding lies what we have chosen to call "1.5 pass decoding". In this case, following the first pass of the decoder, rather than making a complete second pass, only the horizontal codes are utilized and no vertical decoding is attempted.

For simplicity, every bundle was subjected to the 1.5 pass and 2 pass decoding whether or not the first passs registered a failure. This means that the occasional decoding error on the first pass might be picked up by the subsequent pass. In practice, however, the reverse effect turns out to be more likely and we discuss this again in Section 3.3.

The above three strategies were tested on both the bundle and double bundle codes at 15 different lengths each to determine the effects of varying the bundle length. For the bundle code these lengths were from 7 to 21 inclusive and for the double bundle code they were from 14 to 42, in steps of 2 to provide the same information rates for the two codes.

With this background data on the effects of the length of the bundle established, we then considered some alternative lines of attack for the decoder to utilize. The most likely way for the decoder to be unable to decode a single code word is for there to be three bit errors, not all being in the same byte. If these are in three separate bytes, then the vertical decoder will correct them unless there are too many errors in the rest of the bundle. Similarly if the three errors are in two bytes, they will again be corrected vertically unless the code is the single bundle and they also happen to lie in the same vertical codeword (eg. in bytes 1 and 14 horizontally). In this circumstance the bundle will fail because the vertical codeword is unable to locate the even parity error.

We can overcome this effect, however, if the failure of a horizontal codeword is used to blank out that codeword. When this occurs, the power of the erasure decoding ability of the vertical codewords can be used to re-insert the missing data and so allow the bundle to be decoded. If more than one horizontal codeword of the single bundle is erased, however, then all the vertical codes will fail and the bundle will never come through successfully. This suggests the following modification to the original decoding technique: - record the failures of the horizontal decoder and when each packet has been decoded, if exactly one failed, erase that packet prior to decoding vertically. This method allows the vertical decoders the chance to correct a scattering of errors which cause two horizontal words to fail - three single bit errors in each, for example. We will call this <u>Modified Decoding</u>.

In the double bundle, two erasures can be replaced so in this case we erase packets if there are either one or two failures in the horizontal decoding pass. In this way error patterns confined to one or two packets will almost always be correctable so that a brief slip of synchronization in the teletext decoder does not have to lead to a rejected bundle. Again we call this modified decoding.

These modified decoders were then tested on the field data from the sites where the original decoders had been less than 100% successful and each was tested at three different bundle lengths. For the single bundle these were lengths 7, 14 and 21 and for the double bundle they were 14, 28 and 42.

3.2 Bundle Length Effects

To test the decoding strategies at different bundle lengths, encoded bundles were subjected to the error patterns observed at various test sites. In order to do this, the field data were transformed into information concerning the number and locations of errors in each packet as detailed in section 4.2 in our report "Error Correction Schemes for Broadcast Teletext Systems", March 1984. The information was then downloaded into a Radio Shack Colour Computer so that it could be manipulated by the 6809 E microprocessor. For sites with large numbers of errors, the error information had to be subdivided to ensure that it would fit into the 32K of memory available. The microprocessor generated a bundle or double bundle code and then applied the error patterns to this bundle. The particular decoding technique to be tested was then applied to this bundle to see if it could be successfully decoded.

If the decoder reported a failure, this failure was recorded, while if the decoder claimed a success, the decoded bundle was compared with a copy of the original. Only the first 26 bytes of each packet were tested since decoding errors in the check bytes do not affect the data. The check lines were compared, however. If any discrepancies were found then a decoding error was recorded. The bundle was then replaced with a fresh copy and the procedure repeated until all the error information had been used. The number of bundles, failures and errors were then recorded. All of these data were stored on floppy diskettes.

Although we only received error data for rural sites, there was a wide range in the results obtained. At many of the sites, all the decoding strategies managed to remove all of the errors and showed no bundle failures at all in the range considered, so that, trivially, the bundle length had no effect at these sites. At other sites, the errors were concentrated into a small number of packets, which we have called "smeared" packets. In these cases the number of bundles which failed remained constant since success and failure depended on the absence or presence of smeared packets. We note that the modified decoders were much more successful here.

At several sites the error information that was given to us terminated with incomprehensible items and at these sites the data for these last packets was ignored. At site 392 the last 128 packets were all erasures and, since no coding scheme could deal with such a situation, these were discarded.

Rather than display the data obtained for every site we have selected two of the sites at which a large number of the bundles failed and displayed the data in tabular and graphic lay outs. The grand totals over all the sites were also computed to produce an "average" site and the graph for this is shown on the same scale as for the two selected sites to clearly demonstrate how much better the performance is at the remaining sites.

In the histograms, at each length the results are shown for 1,1.5 and 2 pass respectively with the percentage failure shown first, and above this the solid area shows the percentage of bundles containing a decoding error. As expected, the failure rate rises steadily with the bundle length as does the percentage of decoding errors. As Site 301-2 shows most clearly the percentages of decoding errors are much higher for the 1.5 pass decoding than for either 1 or 2 pass decoding.

Failure rates for bundles are not directly comparable with packet failure rates but it is illuminating to investigate the correspondence between packet and bundle failure rates. In particular, for a bundle of 14 packects, if a packet fails with probability p and succeeeds with probability $q = 1 - p$, then the bundle will be correct with probability q^{14} and hence incorrect with probability $1 - q^{14} = 1-(1-p)^{14}$. When p is very small $(p \leq .001)$ this is approximately $14p$ so that the failure rate for the bundles should be compared with the packet rate multiplied at 14. At higher rates the ratio is lower and a bundle failure rate of 50% corresponds to a packet failure rate of 5%.

SITE 301-2

BUNDLE 1 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	2585	97	0	3.75	0.00
[illegible]	2268	101	3	4.45	0.13
[illegible]	2016	104	1	5.16	0.05
10	1814	109	3	6.01	0.17
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	4	7.[illegible]	[illegible]
[illegible]	1395	120	5	8.60	0.36
14	1295	123	6	9.50	0.46
15	1210	123	5	10.17	0.41
16	1134	135	5	11.90	0.44
[illegible]	[illegible]	134	9	12.55	0.84
18	1009	136	7	13.48	0.69
19	955	155	5	16.23	0.52
20	909	153	6	16.83	0.66
21	864	156	8	18.06	0.93

BUNDLE 1.5 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	58	50	[illegible]	[illegible]
10	1814	59	33	3.25	1.82
[illegible]	[illegible]	[illegible]	39	4.05	[illegible]
[illegible]	1516	[illegible]	37	4.44	[illegible]
[illegible]	[illegible]	[illegible]	33	5.23	[illegible]
[illegible]	[illegible]	[illegible]	33	[illegible]	[illegible]
[illegible]	[illegible]	74	35	6.12	2.89
[illegible]	[illegible]	[illegible]	38	7.23	[illegible]
[illegible]	1068	83	34	7.77	3.18
18	1009	79	35	7.83	3.47
19	[illegible]	[illegible]	38	8.90	3.98
20	909	89	34	9.79	3.74
21	864	88	39	10.19	4.51

BUNDLE 2 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	2589	55	8	2.12	0.31
[illegible]	2268	60	7	2.65	0.31
[illegible]	[illegible]	[illegible]	4	[illegible]	0.20
[illegible]	[illegible]	[illegible]	10	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	11	3.58	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	4.04	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	4.15	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	58	13	5.00	1.15
[illegible]	[illegible]	61	17	5.71	1.59
18	1009	60	14	5.95	1.39
[illegible]	[illegible]	[illegible]	15	7.12	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	7.37	[illegible]
[illegible]	864	[illegible]	[illegible]	7.18	[illegible]

TABLE 3.1

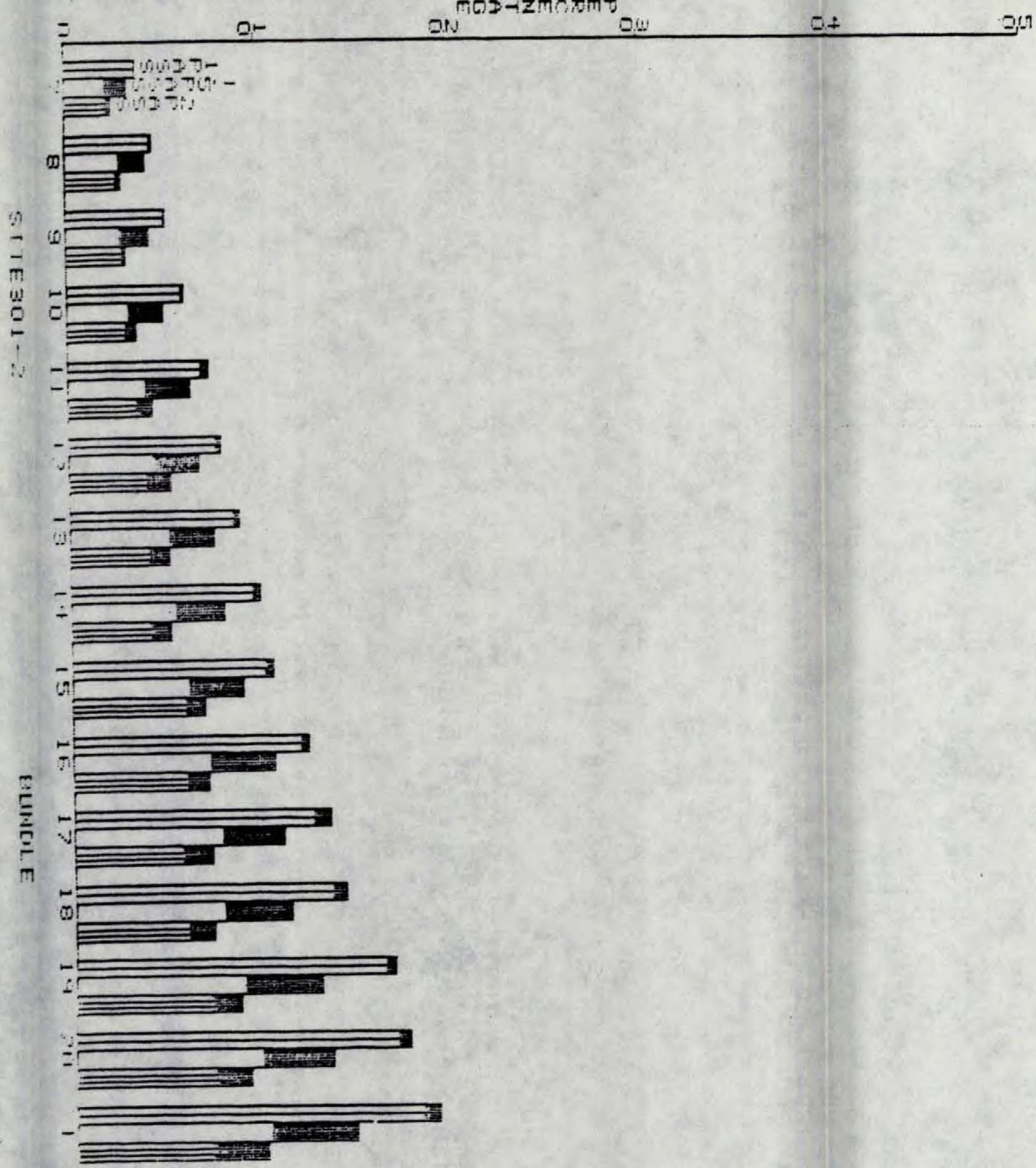


FIGURE 3.1

SITE 301-2

DOUBLE BUNDLE 1 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
14	1295	42	1	3.24	0.08
16	1134	55	1	4.85	0.09
18	1009	61	1	6.05	0.10
[illegible]	909	62	1	6.82	0.11
[illegible]	[illegible]	[illegible]	4	[illegible].99	0.48
[illegible]	[illegible]	[illegible]	3	10.64	[illegible]
26	699	81	3	11.59	0.43
28	648	92	3	14.20	0.46
30	605	88	2	14.55	0.33
32	568	98	5	17.25	0.88
34	535	105	4	19.63	0.75
36	506	101	6	19.96	1.19
38	479	110	7	22.96	1.46
40	456	116	4	25.44	0.88
42	433	112	6	25.87	1.39

DOUBLE BUNDLE 1.5 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	1134	[illegible]	11	3.00	[illegible]
[illegible]	[illegible]	[illegible]	11	2.97	[illegible]
[illegible]	909	[illegible]	11	3.52	[illegible]
[illegible]	[illegible]	[illegible]	11	4.12	[illegible]
[illegible]	[illegible]	42	13	5.28	[illegible]
[illegible]	[illegible]	40	13	5.72	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	6.94	[illegible]
[illegible]	[illegible]	50	[illegible]	8.90	[illegible]
[illegible]	[illegible]	52	15	9.72	2.80
[illegible]	506	53	17	10.47	3.36
[illegible]	[illegible]	62	15	12.94	3.13
40	456	58	16	12.72	3.51
42	433	65	19	15.01	4.39

DOUBLE BUNDLE 2 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
14	1295	3	1	0.23	0.08
16	1134	8	0	0.71	0.00
[illegible]	1009	6	0	0.59	0.00
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	1.21	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	14	1	2.46	0.18
[illegible]	[illegible]	15	2	2.80	0.37
[illegible]	506	17	3	3.36	0.59
[illegible]	[illegible]	18	3	3.76	0.63
[illegible]	[illegible]	25	1	5.48	[illegible]
[illegible]	433	28	5	6.47	1.15

TABLE 3.:

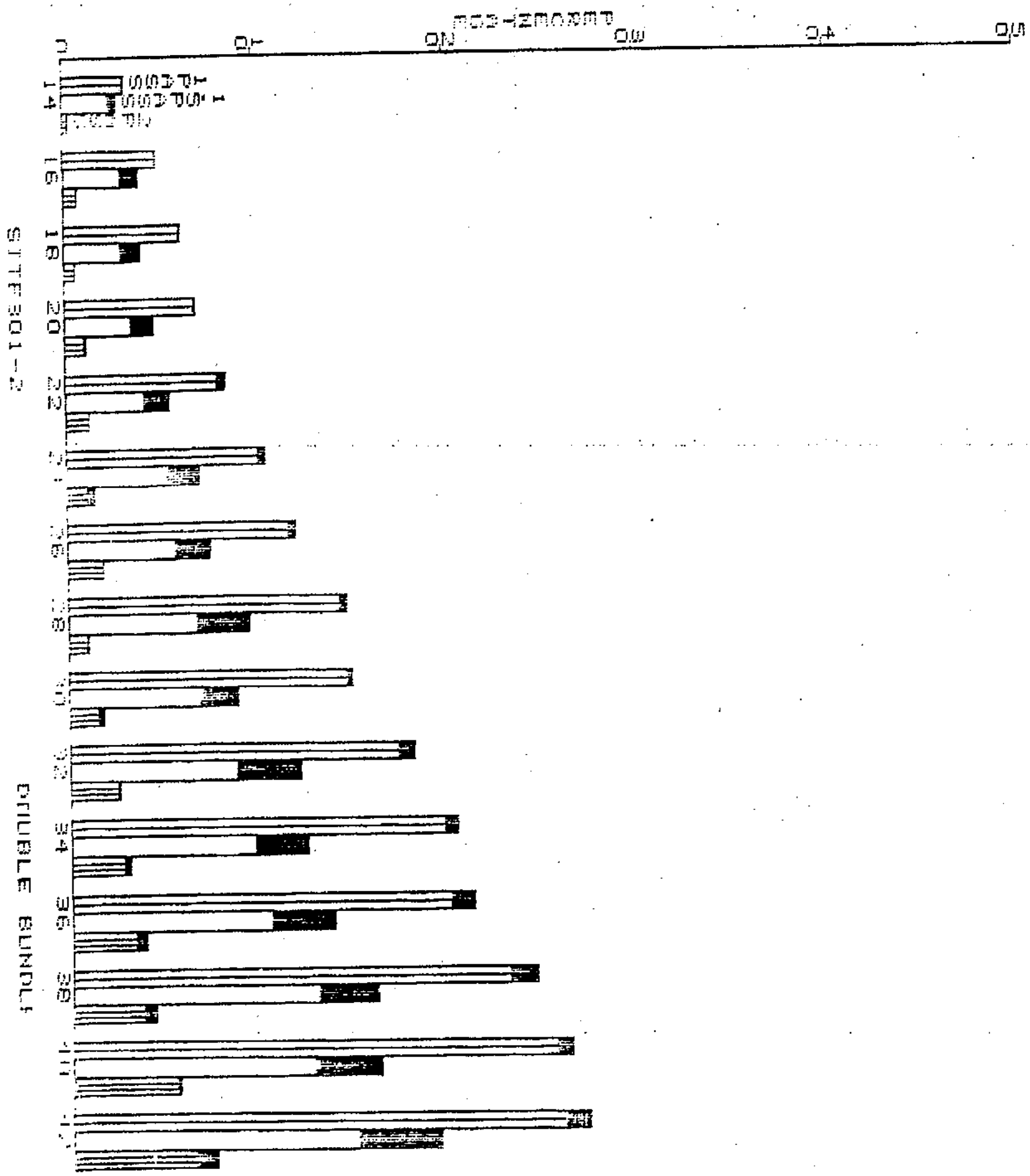


Figure 3.2

SITE 380

BUNDLE 1 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
7	1578	94	1	5.96	0.06
8	1390	103	1	7.46	0.07
9	1233	100	1	8.11	0.08
10	1106	98	1	8.86	0.09
[illegible]	[illegible]	[illegible]	1	[illegible]	0.10
[illegible]	923	101	1	[illegible]	0.11
13	851	101	0	11.87	0.00
14	790	94	1	11.90	0.13
15	739	92	0	12.45	0.00
16	691	93	0	13.46	0.00
17	651	94	0	14.44	0.00
18	616	83	2	13.47	0.32
19	584	85	0	14.55	0.00
20	555	86	1	15.50	0.18
21	528	83	2	15.72	0.38

BUNDLE 1.5 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	2	4.35	[illegible]
[illegible]	[illegible]	[illegible]	1	6.15	0.09
[illegible]	[illegible]	[illegible]	2	6.57	0.29
[illegible]	[illegible]	[illegible]	[illegible]	7.59	[illegible]
[illegible]	[illegible]	[illegible]	0	8.11	0.00
[illegible]	[illegible]	[illegible]	1	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	0	9.34	0.00
[illegible]	[illegible]	70	0	10.13	0.00
[illegible]	[illegible]	66	0	10.14	0.00
18	616	62	0	10.06	0.00
19	584	66	0	11.30	0.00
20	555	64	0	11.53	0.00
21	528	59	0	11.17	0.00

BUNDLE 2 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
7	1578	26	4	1.65	0.25
[illegible]	1390	33	2	2.39	0.14
[illegible]	[illegible]	31	[illegible]	3.51	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	4.38	0.30
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	691	44	2	6.37	[illegible]
[illegible]	651	42	1	6.45	0.15
18	616	49	1	7.95	0.16
[illegible]	584	47	1	8.05	0.17
[illegible]	555	44	2	7.93	0.36
[illegible]	528	45	1	8.52	0.19

TABLE 3.7

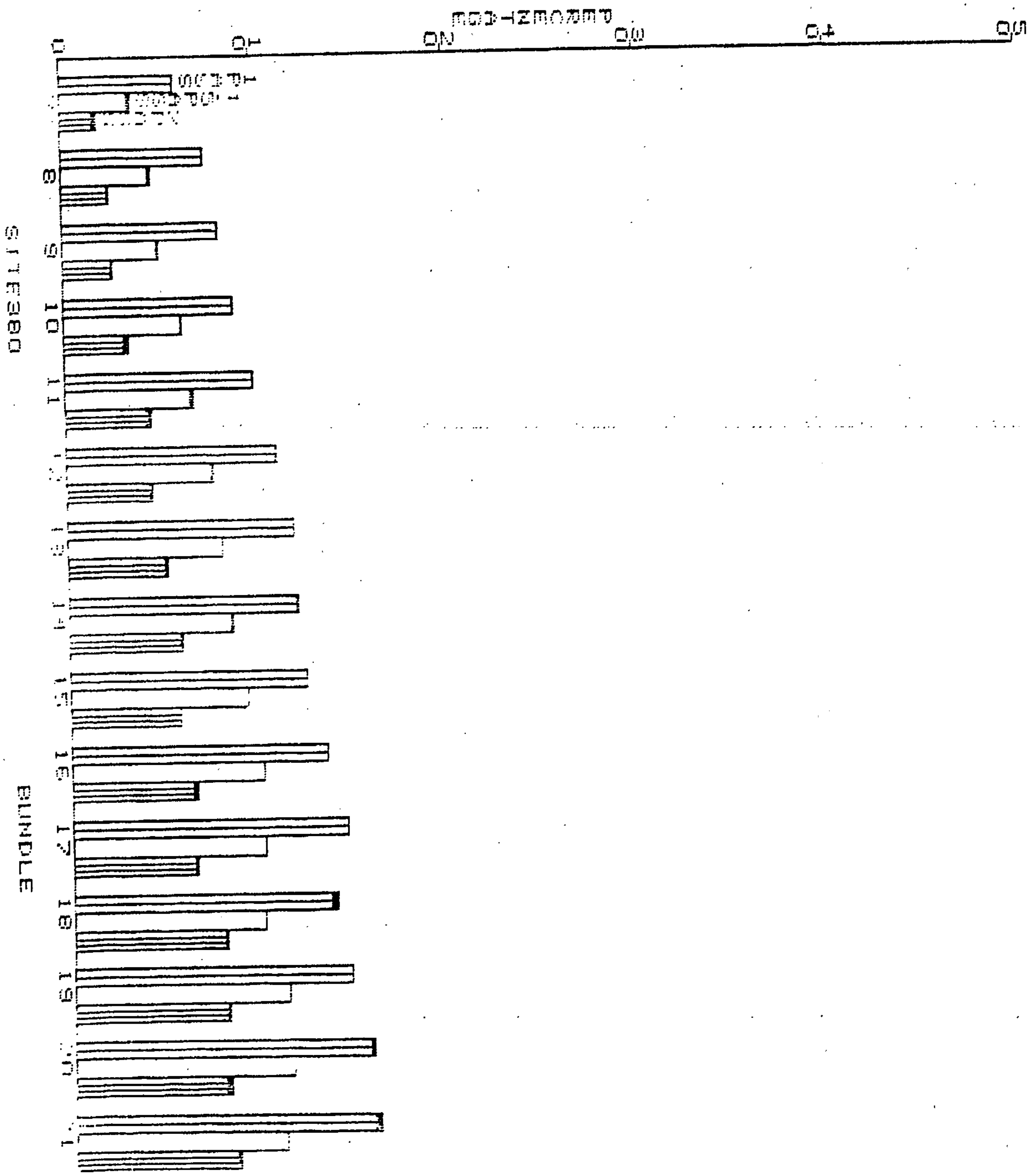


Figure 3..

SITE 380

DOUBLE BUNDLE 1 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
14	790	66	0	8.35	0.00
16	691	68	0	9.84	0.00
18	626	64	0	10.22	0.00
20	555	60	2	10.81	0.36
[illegible]	[illegible]	64	1	12.70	0.20
[illegible]	[illegible]	54	2	11.29	[illegible]
26	426	56	1	13.15	0.23
28	397	58	1	14.61	0.25
30	372	58	0	15.59	0.00
32	347	51	1	14.70	0.29
[illegible]	[illegible]	59	0	18.04	0.00
36	310	50	0	16.13	0.00
38	294	53	0	18.03	0.00
40	279	52	1	18.64	0.36
42	266	50	0	18.80	0.00

DOUBLE BUNDLE 1.5 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	1	5.08	[illegible]
[illegible]	[illegible]	[illegible]	0	[illegible]	[illegible]
[illegible]	[illegible]	40	2	7.21	0.36
[illegible]	504	40	4	7.94	0.79
[illegible]	[illegible]	43	2	9.31	[illegible]
[illegible]	[illegible]	43	2	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	2	10.33	[illegible]
[illegible]	[illegible]	42	3	11.29	[illegible]
[illegible]	[illegible]	43	2	12.39	0.58
34	[illegible]	41	1	12.54	0.31
[illegible]	[illegible]	38	2	12.26	0.65
[illegible]	294	37	2	12.59	0.68
40	279	38	1	13.62	0.36
42	266	37	1	13.91	0.38

DOUBLE BUNDLE 2 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
14	790	20	3	2.53	0.38
[illegible]	691	22	2	3.18	0.29
[illegible]	[illegible]	[illegible]	1	4.31	[illegible]
[illegible]	[illegible]	23	3	4.14	[illegible]
[illegible]	[illegible]	28	2	5.56	[illegible]
[illegible]	[illegible]	[illegible]	3	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	1	7.81	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	31	1	8.93	[illegible]
[illegible]	[illegible]	33	0	10.09	0.00
[illegible]	[illegible]	27	1	8.71	0.32
[illegible]	[illegible]	27	0	9.18	0.00
[illegible]	[illegible]	[illegible]	1	10.39	[illegible]
[illegible]	[illegible]	27	2	10.15	[illegible]

TABLE 3.

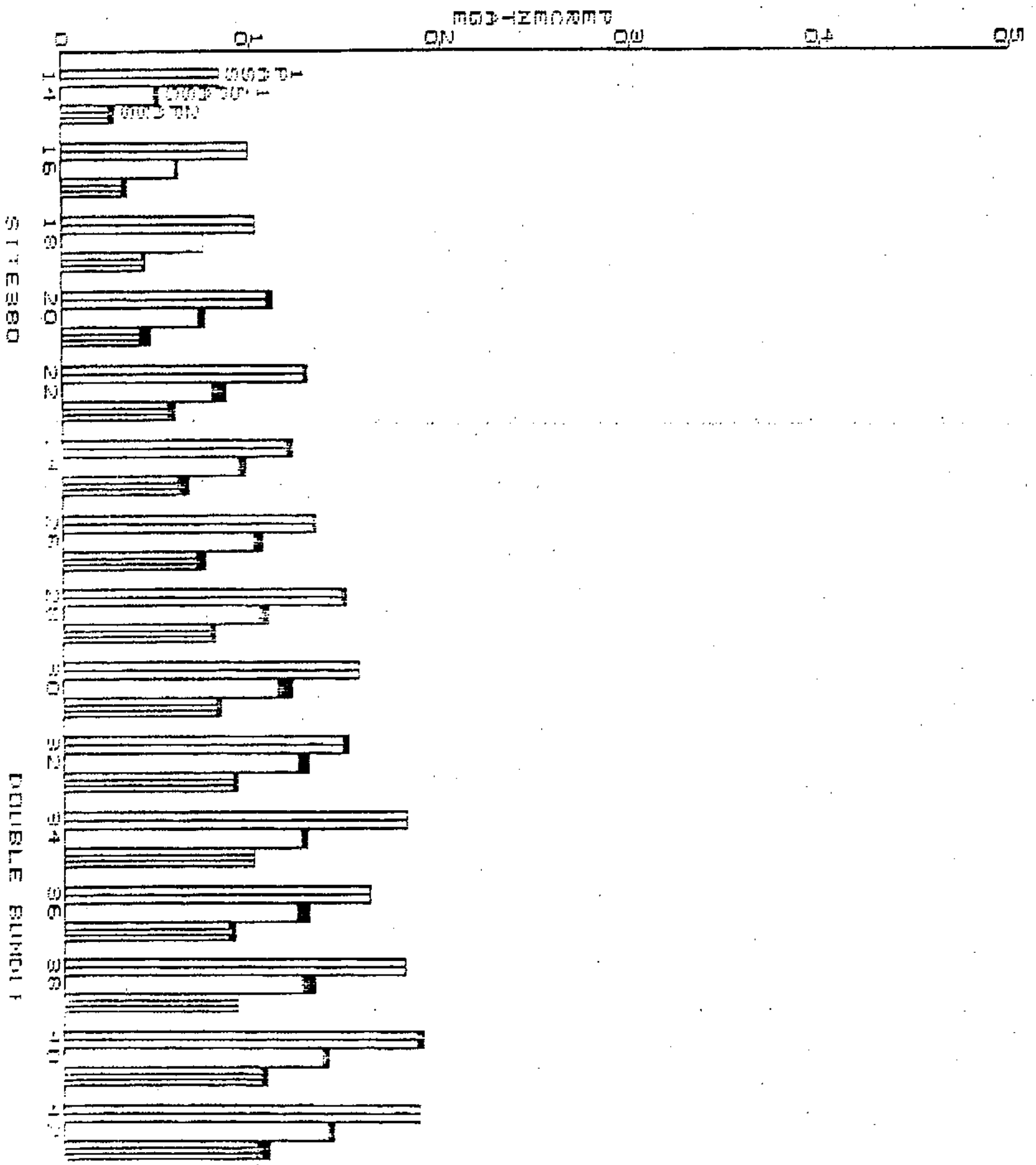


Figure 3.

SITE AVERAGE

BUNDLE 1 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
7	116020	902	17	0.78	0.01
8	102522	940	17	0.92	0.02
9	90253	946	14	1.05	0.02
10	81317	974	19	1.20	0.02
11	[illegible]	[illegible]	24	1.35	[illegible]
12	[illegible]	[illegible]	[illegible]	1.47	0.03
13	62584	1016	15	1.62	0.02
14	58017	1012	19	1.74	0.03
15	54159	1035	20	1.91	0.04
16	50779	1056	21	2.08	0.04
17	47791	1046	23	2.19	0.05
18	45145	1061	23	2.35	0.05
19	42767	1083	22	2.53	0.05
20	40632	1091	21	2.69	0.05
21	38524	1101	22	2.86	0.06

BUNDLE 1.5 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	[illegible]	[illegible]	131	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	728	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	92	0.95	0.11
[illegible]	[illegible]	771	102	1.04	0.14
[illegible]	[illegible]	775	100	1.14	[illegible]
[illegible]	[illegible]	[illegible]	101	1.25	[illegible]
[illegible]	[illegible]	[illegible]	101	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	95	1.52	0.18
[illegible]	[illegible]	[illegible]	104	1.63	0.20
[illegible]	47791	814	91	1.70	0.19
18	45145	824	103	1.83	0.23
[illegible]	42767	841	96	1.97	0.22
20	40632	852	92	2.10	0.23
21	38524	849	94	2.20	0.24

BUNDLE 2 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	116020	619	36	0.53	0.03
[illegible]	102522	636	34	0.62	0.03
[illegible]	[illegible]	[illegible]	[illegible]	0.71	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	1.05	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	55	[illegible]	[illegible]
[illegible]	47791	682	42	1.43	0.09
18	45145	685	48	1.52	0.11
19	42767	693	43	1.62	0.10
20	40632	702	51	1.73	0.13
[illegible]	38524	697	58	1.81	0.15

TABLE 3.

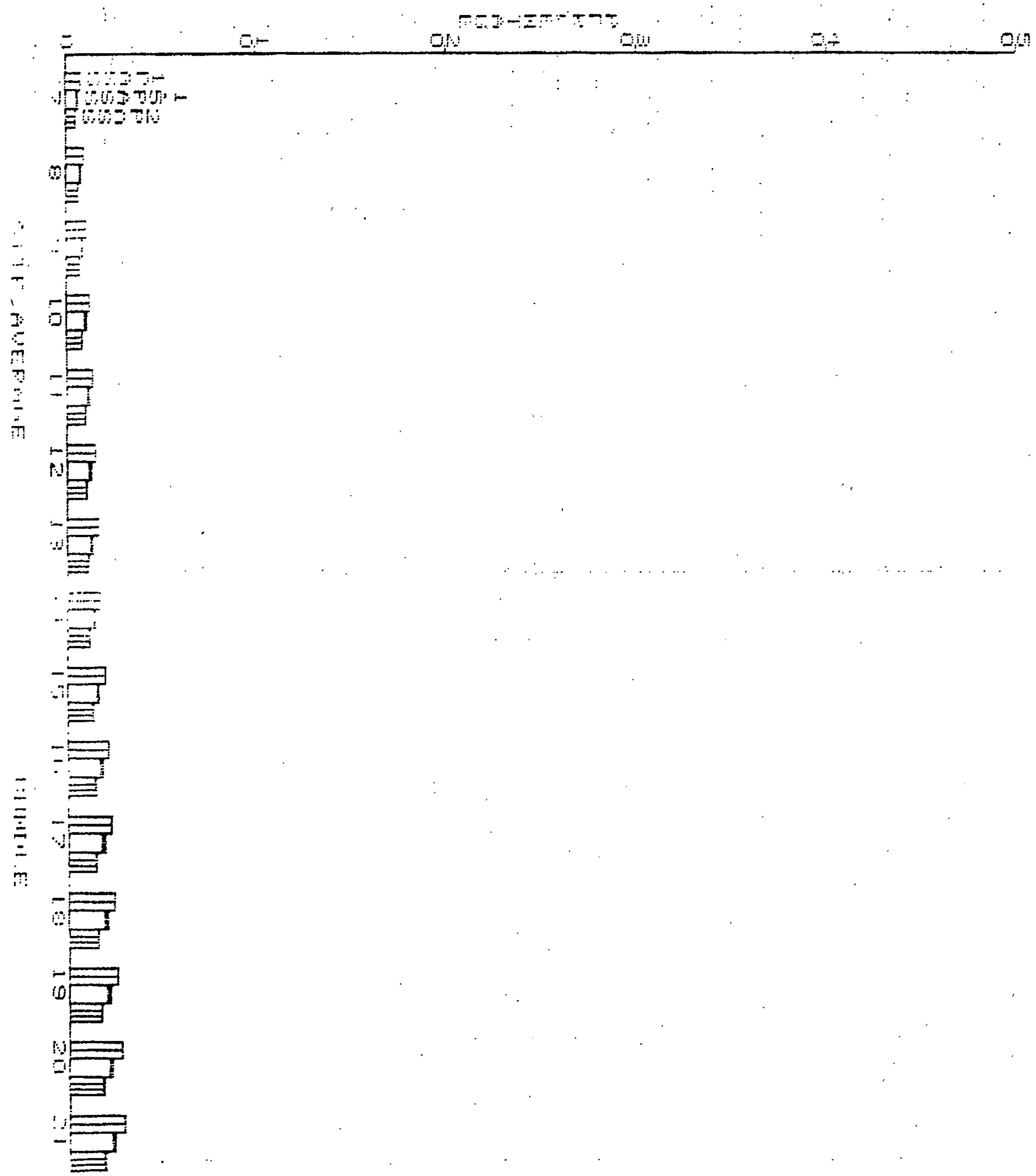


Figure 3.5

SITE AVERAGE

DOUBLE BUNDLE 1 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
14	58046	452	6	0.78	0.01
16	50780	487	7	0.96	0.01
18	45154	515	6	1.14	0.01
[illegible]	40533	521	7	1.28	0.02
[illegible]	[illegible]	[illegible]	12	1.52	[illegible]
[illegible]	[illegible]	594	11	1.75	0.03
26	31259	581	9	1.86	0.03
28	29030	631	11	2.17	0.04
30	27100	624	14	2.30	0.05
32	25409	623	15	2.45	0.06
[illegible]	23914	666	7	2.78	[illegible]
[illegible]	22593	661	12	2.93	0.05
38	21405	660	18	3.08	0.08
40	20333	668	14	3.29	0.07
42	19368	683	17	3.53	0.09

DOUBLE BUNDLE 1.5 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	51	0.94	[illegible]
[illegible]	[illegible]	482	56	1.07	[illegible]
[illegible]	40533	492	51	1.21	0.13
[illegible]	[illegible]	499	61	1.35	0.17
[illegible]	33865	534	61	1.58	0.18
[illegible]	[illegible]	[illegible]	[illegible]	1.66	0.19
[illegible]	[illegible]	[illegible]	[illegible]	1.87	0.22
[illegible]	27100	[illegible]	54	2.02	0.20
[illegible]	25409	565	62	2.22	0.24
[illegible]	23914	560	61	2.34	0.25
[illegible]	22593	582	59	2.58	0.26
[illegible]	21405	568	55	2.65	0.30
40	20333	571	64	2.81	0.31
42	19368	590	60	3.05	0.31

DOUBLE BUNDLE 2 PASS DECODING

LENGTH	BUNDLES	FAILS	ERRORS	%FAIL	%ERROR
14	58046	171	10	0.29	0.02
16	50780	194	7	0.38	0.01
[illegible]	45154	[illegible]	7	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	10	0.61	[illegible]
[illegible]	[illegible]	[illegible]	16	0.69	[illegible]
[illegible]	[illegible]	[illegible]	9	0.78	[illegible]
[illegible]	[illegible]	[illegible]	9	0.85	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	0.95	[illegible]
[illegible]	25409	264	[illegible]	1.04	[illegible]
34	23914	275	12	1.15	[illegible]
[illegible]	[illegible]	280	15	1.24	0.07
[illegible]	[illegible]	289	14	1.35	[illegible]
[illegible]	[illegible]	297	14	1.46	0.07
[illegible]	19368	[illegible]	20	1.55	[illegible]

TABLE 3.

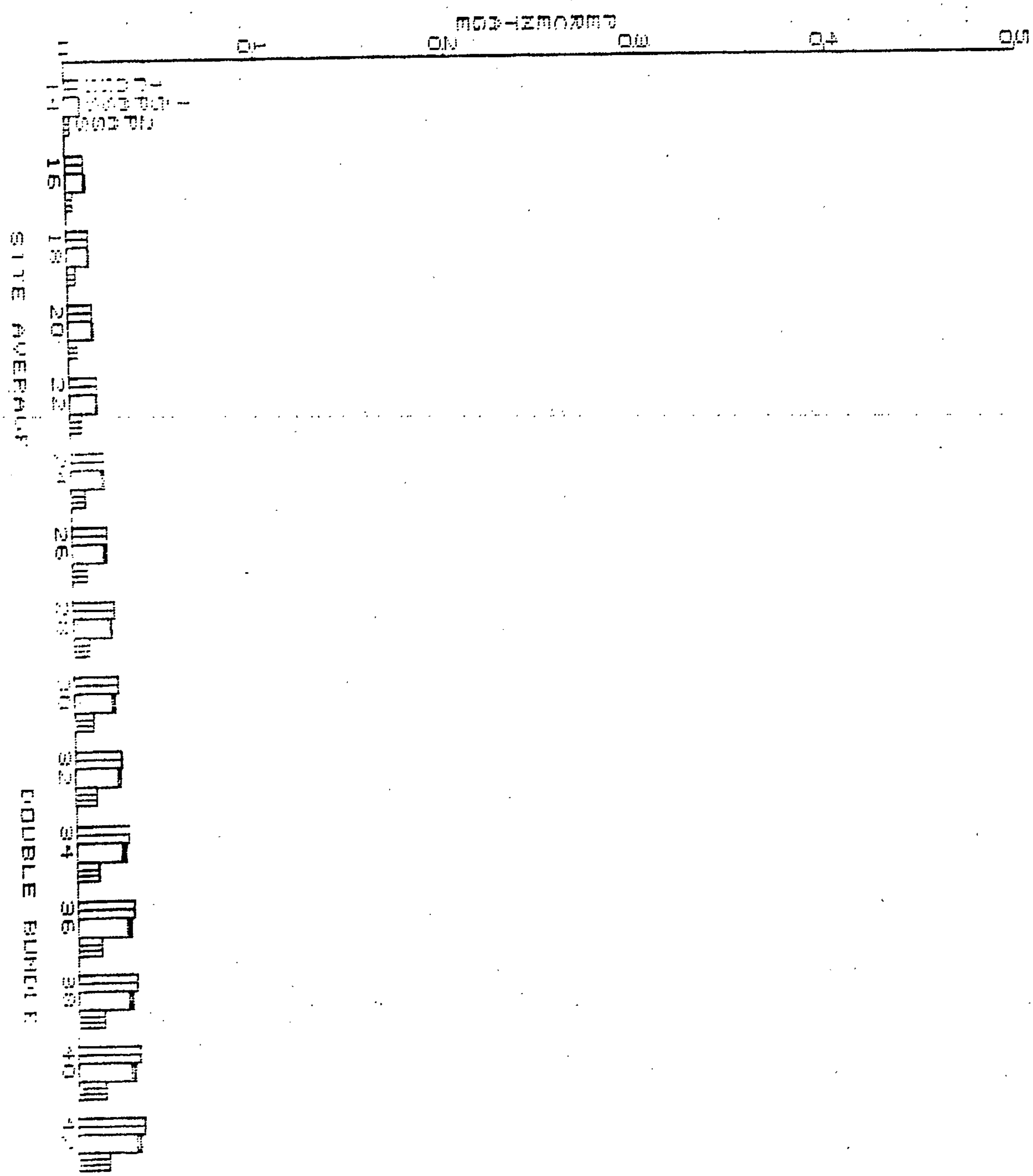


Figure 3.

As the bundle length increased both the failure rate and the information rate increase. For a bundle of length 7 the information rate is 0.796 and at length 14 it is 0.884.* Here the information rate includes onlv the 28 bytes in the packet and ignores the packet headers. For the "average" site we see that the failure rate climbs faster than the information rate. Consequently, it is not advantageous to lengthen the bundles to increase the information rate since the failures will result in no gain.

3.3 Results of Strategies: Recommendations

The modified decoders, which erase one or two packets where the horizontal code fails, were tested at three different bundle lengths. For the single bundle code these lengths were 7, 14 and 21 while the double bundle was tested at 14, 28 and 42. Very similar results were observed at these lengths so we will confine our discussion to the shortest length for each code. Because the modification to the decoder was made after the esting process had begin, the first few sites we received were not tested with the modified decoder. However, we have no reason to suspect that they would change the relative figures. Table 3.7 shows the results obtained.

Table 3.7

	1 PASS				2 PASS			
	SINGLE		DOUBLE		SINGLE		DOUBLE	
	FAIL	ERROR	FAIL	ERROR	FAIL	ERROR	FAIL	ERROR
ORIGINAL	783	16	405	5	546	25	163	6
MODIFIED	471	40	364	26	224	73	160	49

For the single bundle there is a dramatic improvement in the number of failures at both the 1 pass and the 2 pass decoding. This is offset, however, by the marked increase in the number of decoding errors detected. If the bundle code were to be selected this trade off would have to be taken into account. Is it better for the user to wait for retransmission more frequently or should erroneous data be allowed through more often?

In the case of the double bundle code the situation is much clearer. Here the improvement is slight for single pass decoding and marginal for two pass decoding. The deterioration in the rate of erroneous decoding is also very obvious and causes us to reject this strategy of decoding. Out of the four possible schemes shown in Table 3.7, the strategy of choice is clearly two pass decoding of the double bundle without the modification.

Having decided that two pass decoding is superior to single pass decoding, the next logical question to ask is whether three or more passes should be tried. Accordingly we picked three sites and tested the effect of running the decoder three times over each bundle. Using length 7 for the bundle and length 14 for the double bundle, we counted the decoding failures and errors at Sites 380, 386 and 387.

	Bundle		Double Bundle	
Passes	Fails	Errors	Fails	Errors
1	243	2	171	2
2	158	5	76	3
3	150	4	58	2

Table 3.8

At Sites 386 and 387 there was little change in going from 2 to 3 passes and we can draw few conclusions from such a small amount of data. There was an improvement, however. Although the machine language programme on the 6809 E microprocessor spent quite a lot of its time inserting the errors and then comparing the decoded message with the original, it was noticable to us that the process of 3 pass decoding was much slower than 1 pass. Because this was noticable, 3 pass decoding is not recommended.

Since 2 pass decoding was also slower than 1 pass because every bundle is decoded twice, we ran one final test on a multi-pass strategy. For this test, when the bundle finished decoding the number of vertical codewords which failed was saved. If this is zero, the decoder moves to the next bundle. If it is not zero, the decoder re-tries the bundle. The number of failures is again tested and, if zero, we go to the next bundle. If the number of failures is constant, again we go to the next bundle, but if any improvement is detected we go back to the re-try procedure.

This multi-pass procedure was tried on the same three sites as the 3 pass decoder and was much faster while producing almost identical results: 2 less fails, 1 more error for the bundle and 3 less fails, 3 more errors for the double bundle. We then made further tests on two more sites where a large number of failures had been observed, namely sites 302-1 and 370, plus site 377 where few failures had occured. The results are shown in table 3.9.

Table 3.9

Number of decoding failures and errors at 6 sites.

	Bundle		Double Bundle	
Passes	Fails	Errors	Fails	Errors
1	487	15	320	4
2	320	20	121	5
Multi	307	24	99	9

In conclusion, after examing data from forty sites, all of which are rural, the code and decoding strategy which displayed the best performance was the double bundle code with 2 pass (or multi-pass) decoding. At the sites tested the multi-pass decoder was faster so that, although it gave a slightly higher decoding error rate (we estimate 0.04% of the bundles would contain undetected errors) the multi-pass decoder should be used. At bundle length 14, this would give a failure rate of 0.24% of the bunldes.

Chapter 4. The Distribution of Errors in the Channel

4.1 The Channel

The field tests collect data from broadcast teletext signal multiplexed as part of a standard broadcast TV channel. One teletext packet is sent as one line in the TV channel hence packets are broadcast at a rate of 60 per second. Within the packets bits are transmitted at the rate of 5.7676×10^6 bits per second. Thus the packets are 264 bits of data separated in the channel by the equivalent of more than 5×10^6 bits of silence. Thus individual packets are effectively separated in the channel. We will look at this point again in Section 4.6 below.

The packets that are sent have the form of 28 bytes the first three of which are fixed at 02, 09, 00 in hexadecimal. The next 2 bytes are a counter. The remaining 23 bytes are 8 bit segments of a pseudo-noise sequence.

The signal is received and digitized by a Norpak Mark IV decoder. Erroneous bits are logged.

We have used data from four transmitters.

1)	Camp Fortune, Quebec	5 sites
2)	Saskatoon, Saskatchewan	17 sites
3)	North Battleford, Saskatchewan	5 sites
4)	Stranraer, Saskatchewan	10 sites

Each site is identified by a site number and a site location. The geographical distributions are shown on the accompanying maps, Figures 4.1, 4.2 and 4.3.

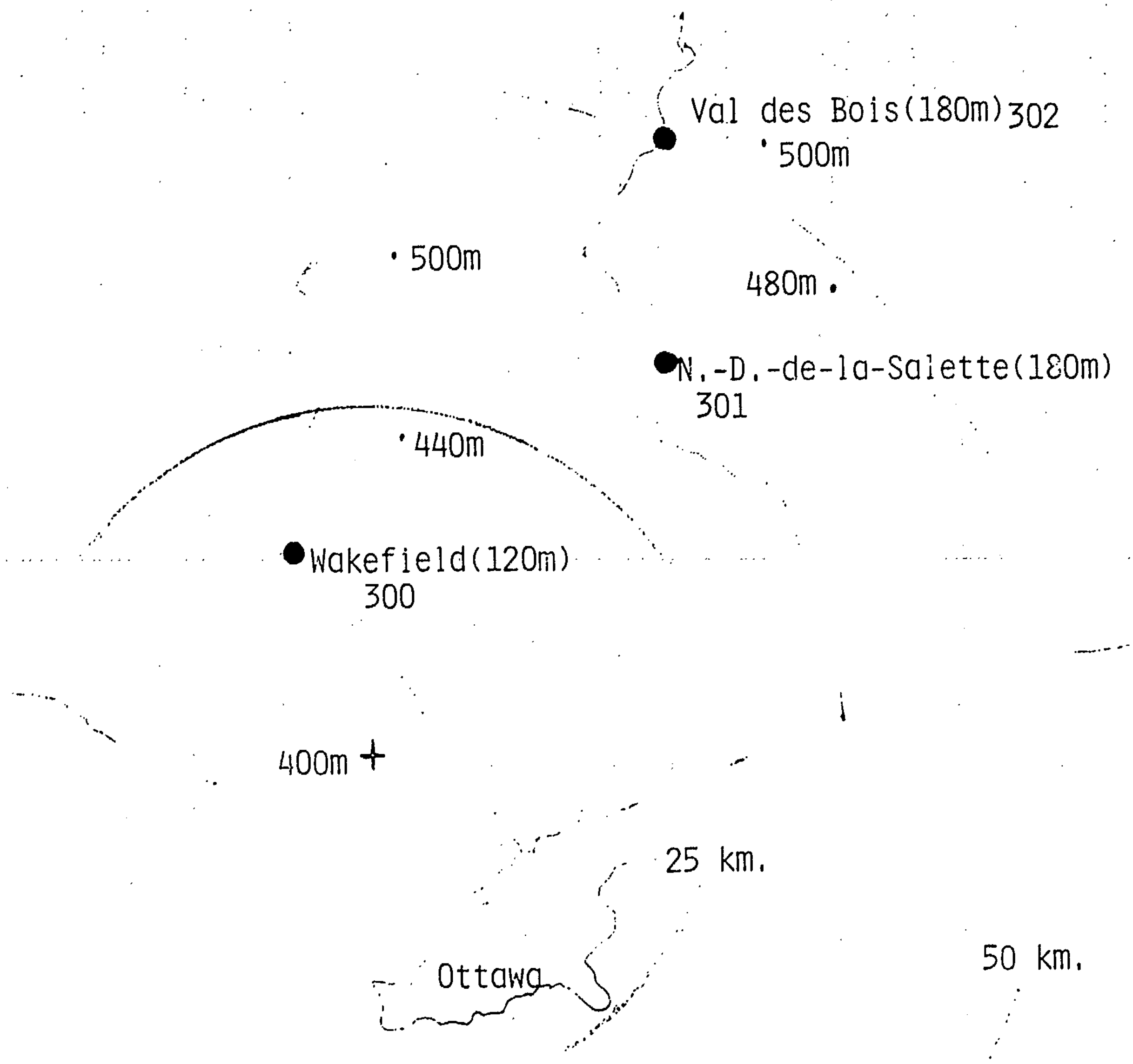


Figure 4.1 Fortune-Gatineau-Lievre sites

+ = Camp Fortune transmitter

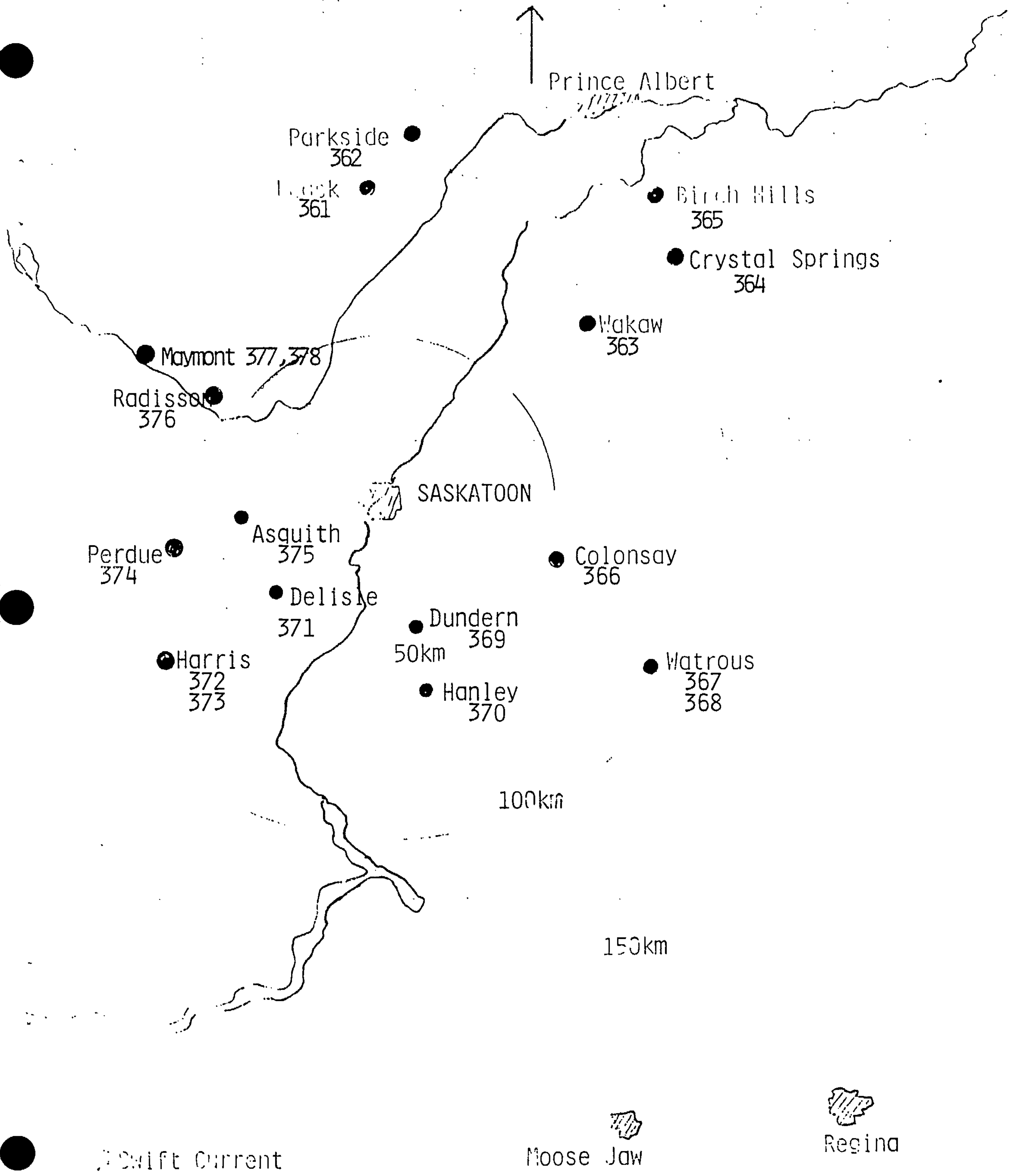


Figure 4.2: Saskatoon Sites

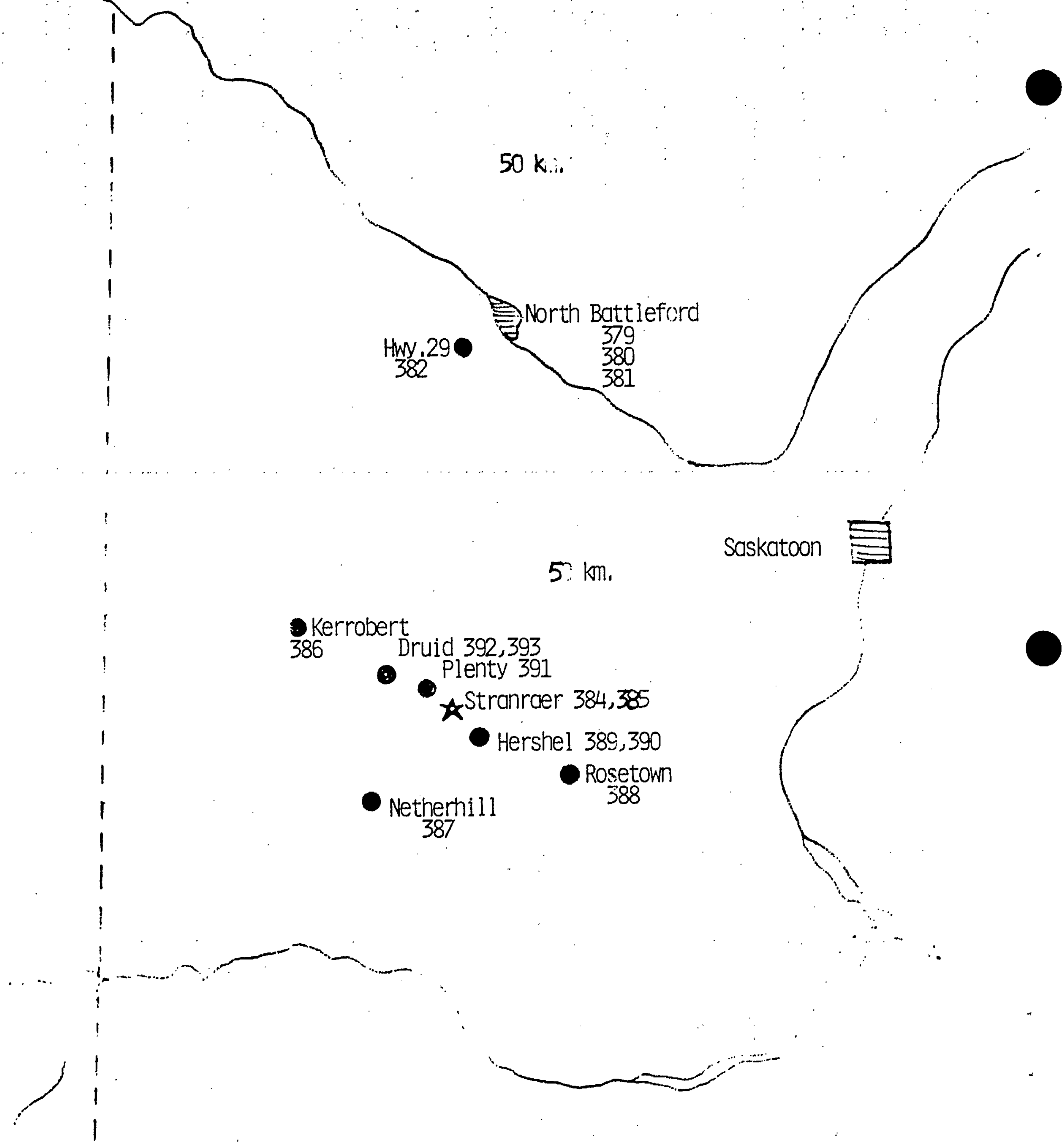


Figure 4.3: North Battleford and Stranraer Sites

4.2 Bit Error Rate

The most commonly used statistic of a digital communication channel is the overall bit error rate (BER) . We have calculated this at each site. At a number of sites where were smeared packets, that is packets with at least 20 erroneous bytes. We will see below in Section 4.3 that these smeared packets are best thought of as the result of a near miss of sychronization rather than a burst of noise. We take these packets out of the bit error rate calculation. Also at some sites there are long quiet stretches and these are also removed from the bit error rate. The result is an Adjusted Bit Error Rate (ABER) which is the rate of bit errors in non-smeared packets. A segment of 1000 errorless packets is dropped from the ABER calculation. We have used this ABER in Chapter 2 in assessing code performance.

We have plotted in Figure 4.4 the adjusted BER against distance to the transmitter. We see that ABER is most sensitive to the transmitter at which tests were made. There appear to be two series of sites. One series of noisy sites includes the Camp Fortune, North Battleford and some of the Stranraer sites. The other series includes all but one of the Saskatoon sites and the remaining Stranaer sites.

The first series show an ABER increasing with distance to the transmitter (although North Battleford is really just a very nasty place to receive a teletext signal). The values of ABER here are from 2e-4 to 5e-3.

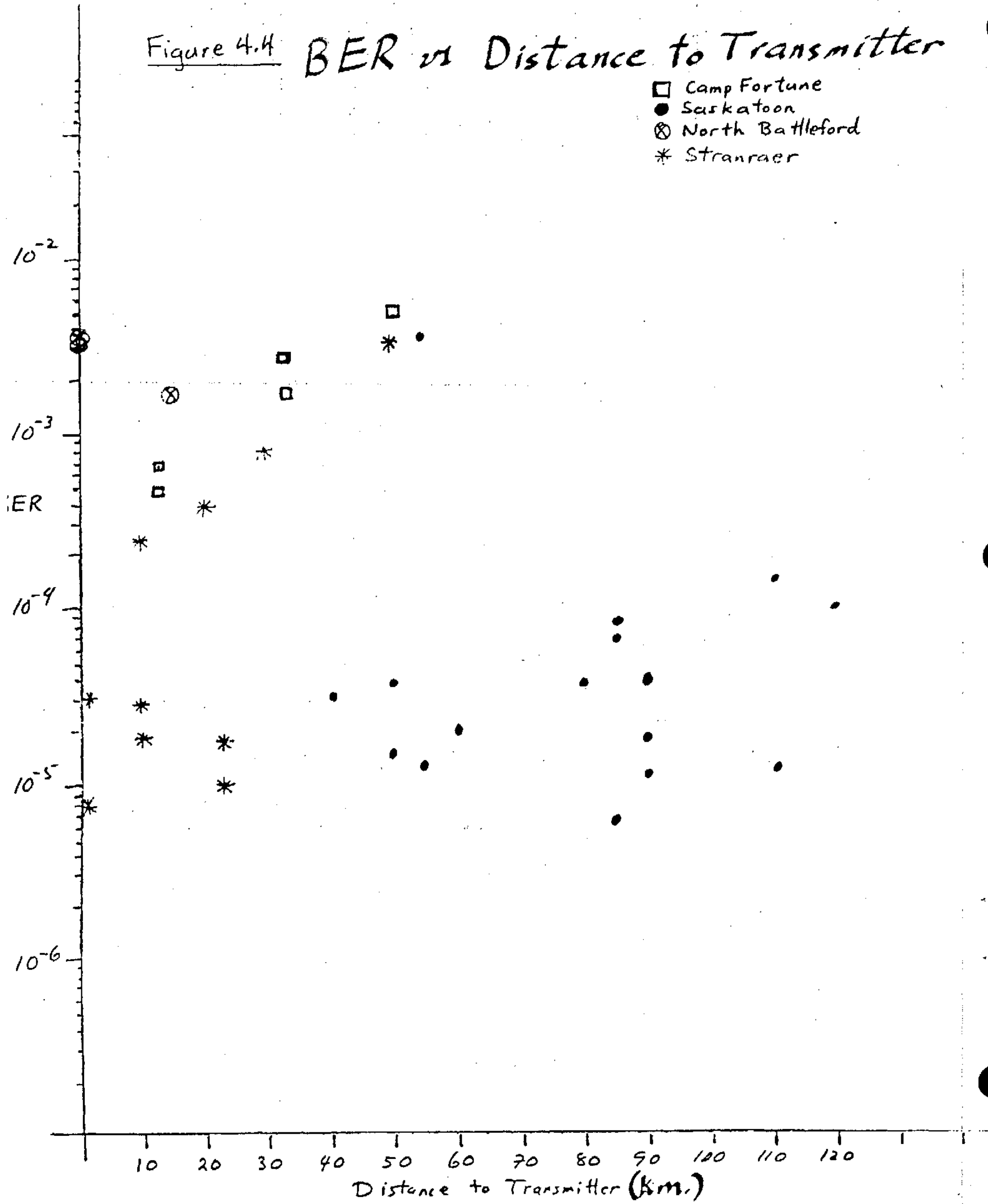


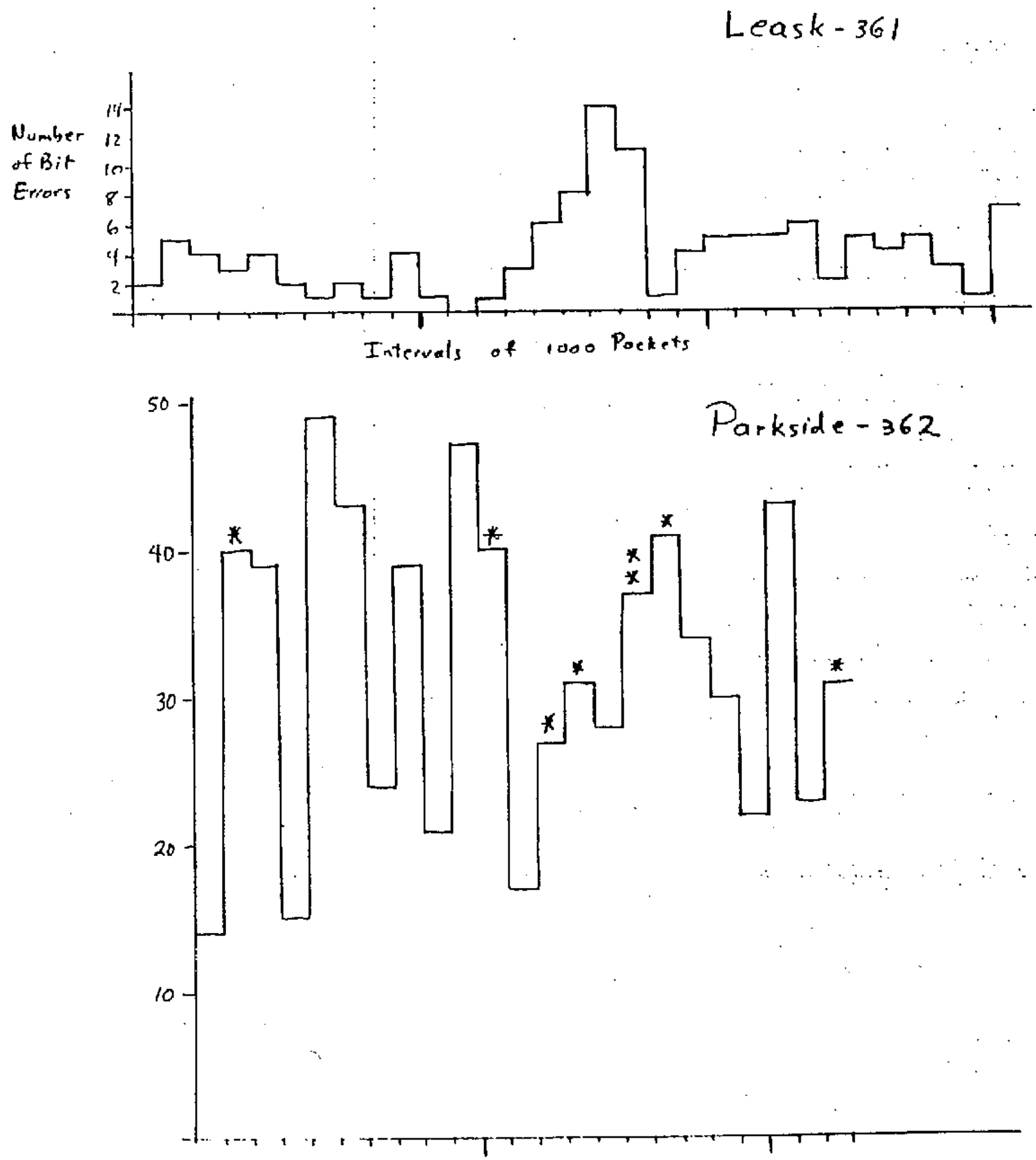
Figure 4.4 BER vs Distance to Transmitter
Camp Fortune
Saskatoon
North Battleford
Stranraer
10^-2
10^-3
BER
10^-4
10^-5
10^-6
10
20
30
40
50
60
70
80
90
100
110
120
Distance to Transmitter (km.)

The second series shows little trend with increasing distance to the transmitter. ABER runs from 8e-6 to 2e-4 . It should be noted that the topography at Saskatoon is only mildly undulating while the Camp Fortune sites are in deep valleys with high hills between them and the transmitter.

The other noticable effect on the ABER is its local variation with time. The channel was divided into segments of 1000 packets and the ABER was calculated for each. At 13 sites there was a clear variation in this local ABER. Actual numbers of bit errors have been plotted for some of the sites in Figure 4.5. Sites 366, 367 and 368 for example had 4000 clean packets (about 1 minute on the channel) then noise until the test stopped. The variations observed are several minutes long so should not be called bursts of noise, but variations in the channel.

Figure 4.5

Number of Bit Errors in Segments of 1000 Packets – Smeared Packets Excluded.

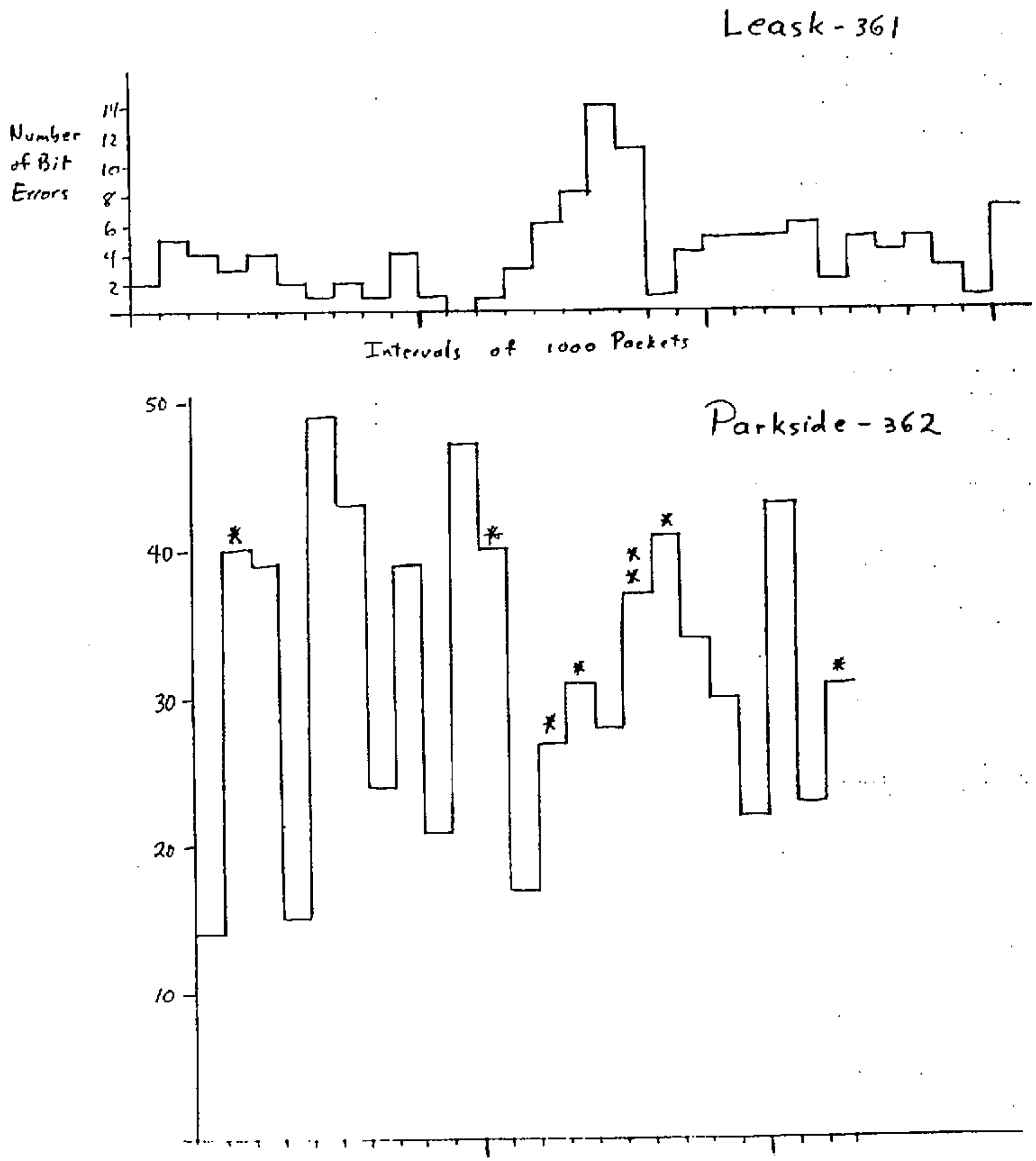


* ≡ 1 smeared packet

1 minute = 3600 packets

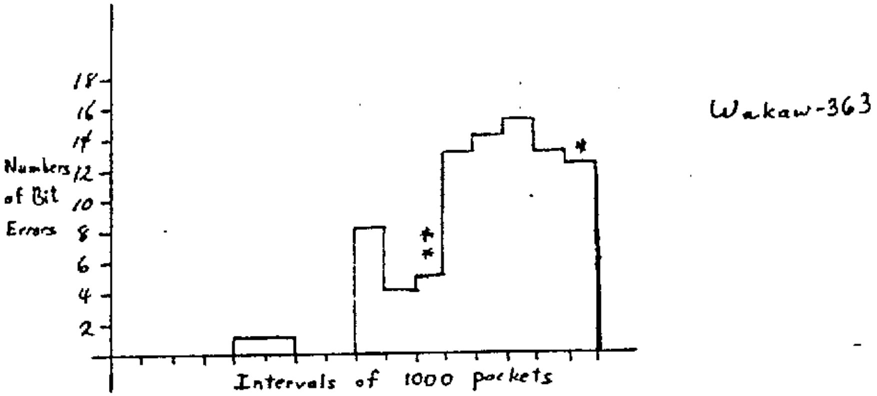
Figure 4.5

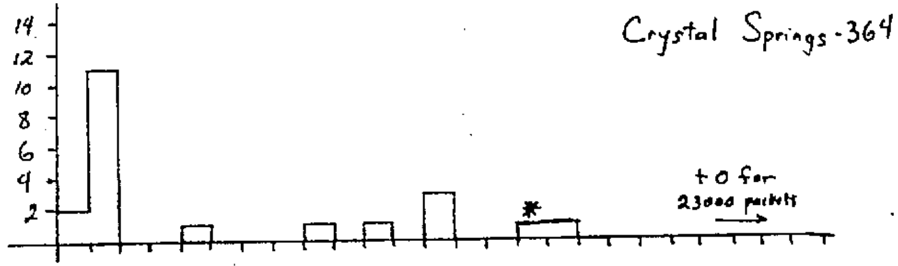
Number of Bit Errors in Segments of 1000 Packets - Smeared Packets Excluded.

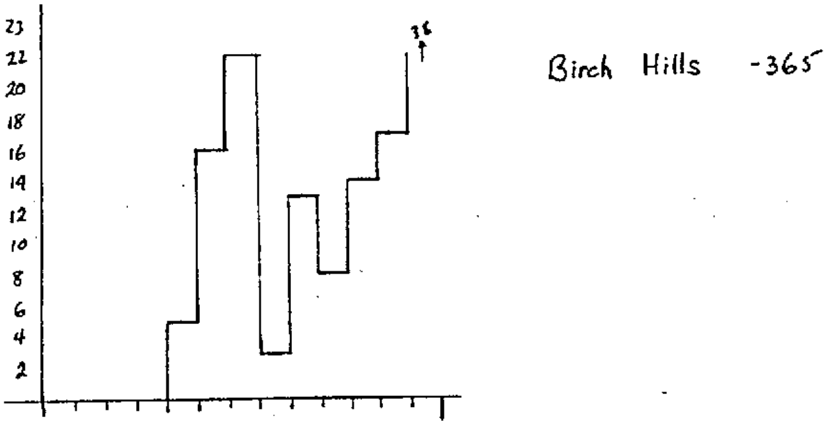


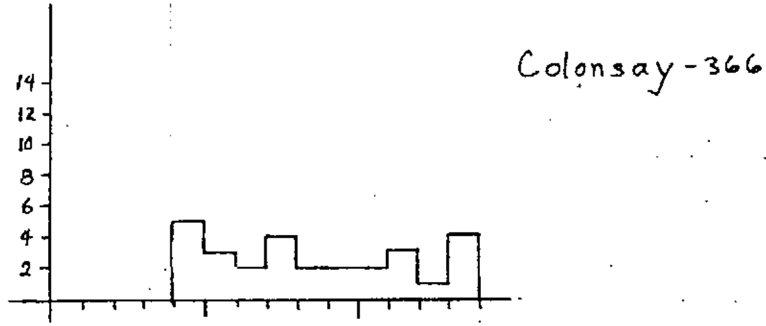
⁕ ≡ 1 smeared packet

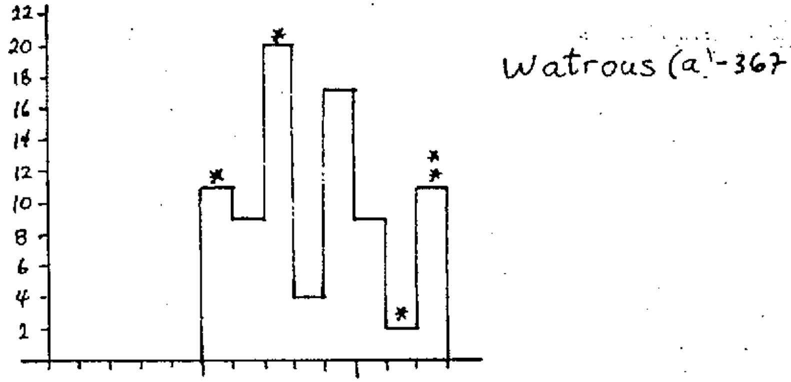
1 minute = 3600 packets

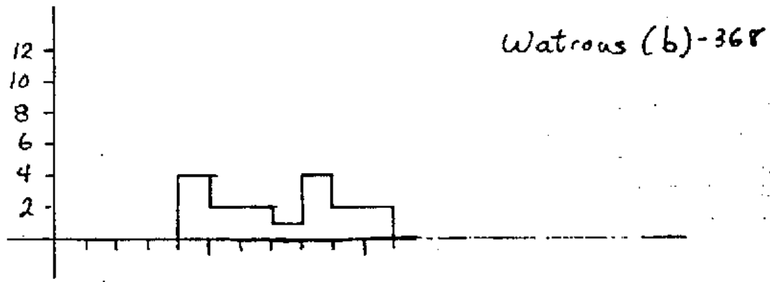

Wakaw-363
Numbers of Bit Errors
18
16
14
12
10
8
6
4
2
Intervals of 1000 packets

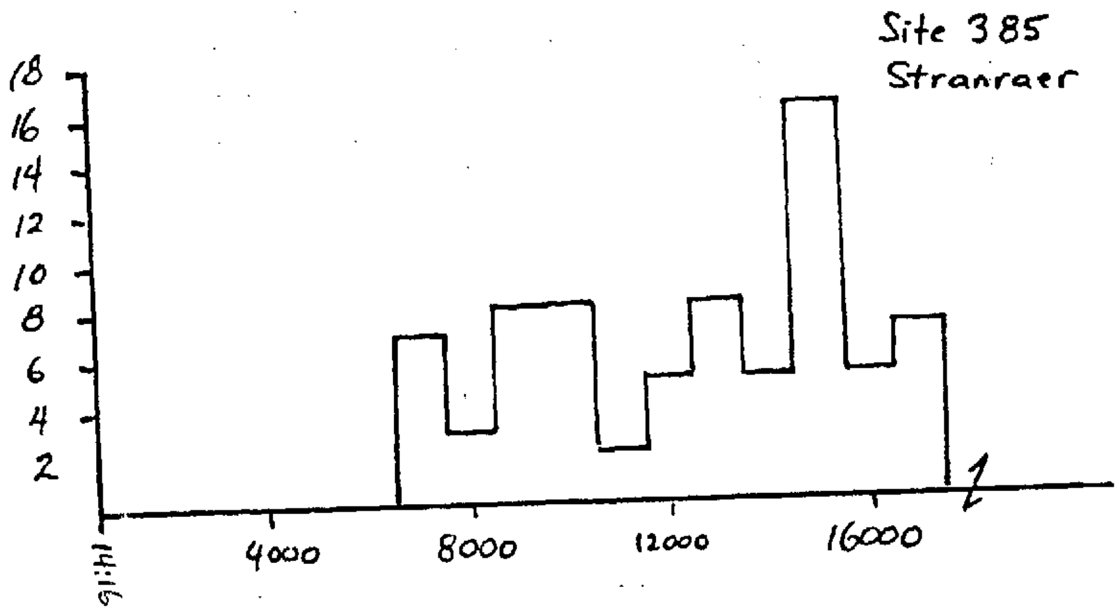

Crystal Springs-364
14
12
10
8
6
4
2
+0 for 23000 packets

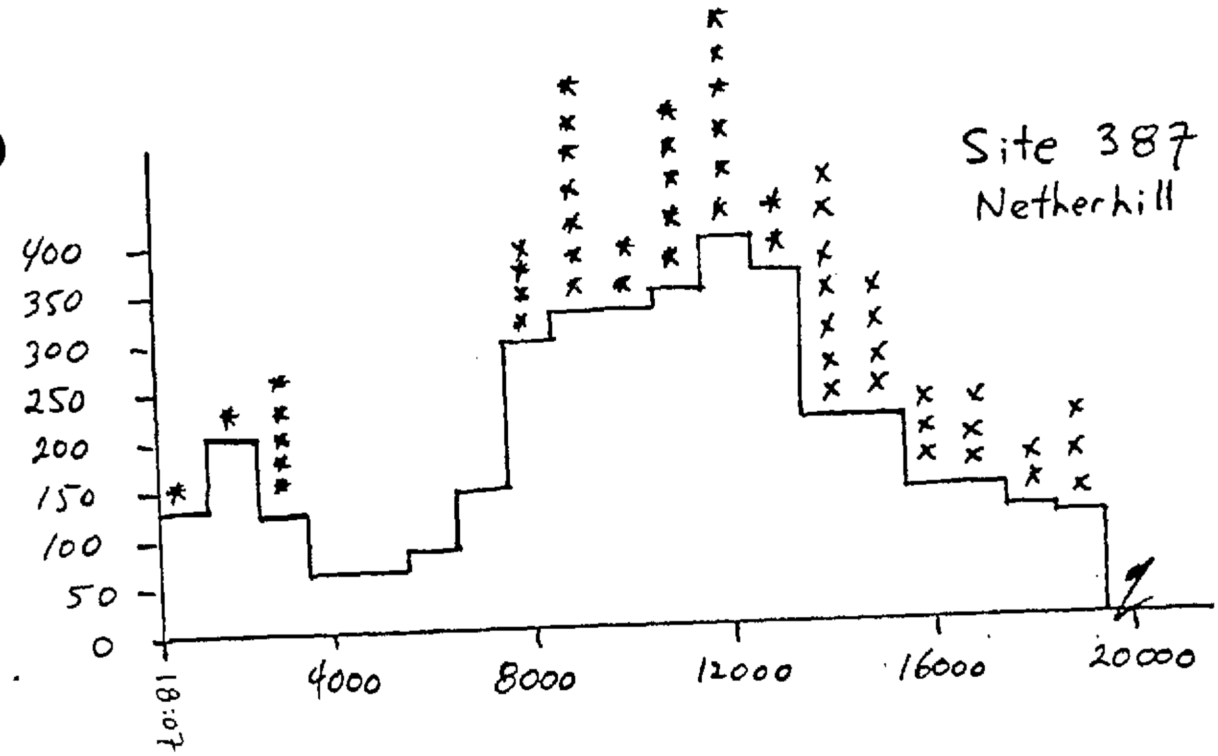

Birch Hills -365
23
22
20
18
16
14
12
10
8
6
4
2
36

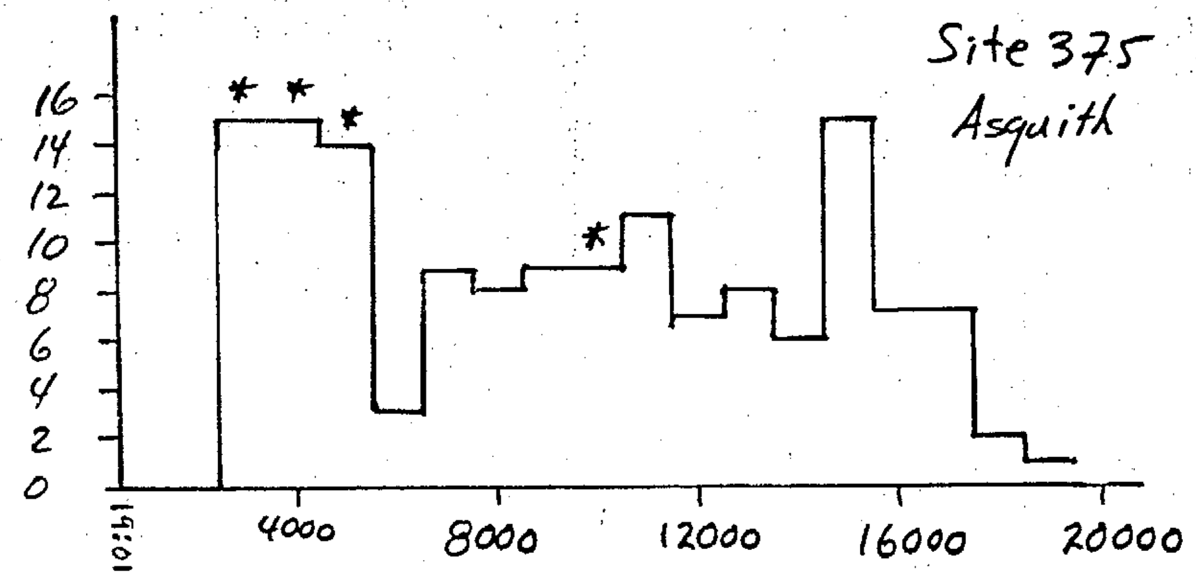

Colonsay-366
14
12
10
8
6
4
2

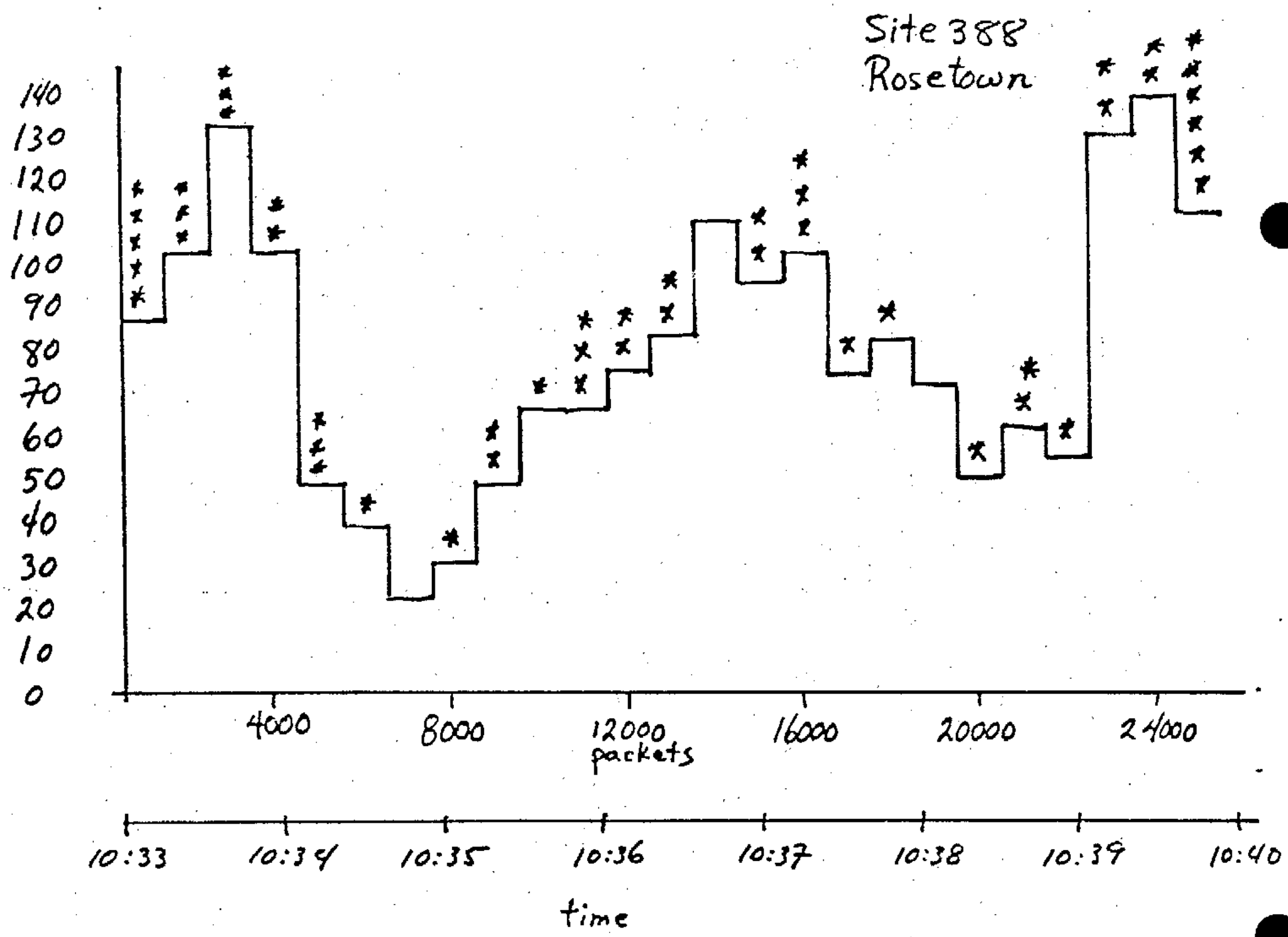

Watrous (a)-367
22
20
18
16
14
12
10
8
6
4
2


Watrous (b)-368
12
10
8
6
4
2


Site 385
Stranraer
18
16
14
12
10
8
6
4
2
4000
8000
12000
16000
14:16


Site 387
Netherhill
400
350
300
250
200
150
100
50
0
4000
8000
12000
16000
20000
18:07


Site 375
Asquith
16
14
12
10
8
6
4
2
0
10:41
4000
8000
12000
16000
20000


Site 388
Rosetown
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
4000
8000
12000
packets
16000
20000
24000
10:33
10:34
10:35
10:36
10:37
10:38
10:39
10:40
time

4.3 Smeared Packets.

We define a smeared packet to be a packet with at least 20 of its 28 bytes containing errors. This is an arbitrary division but in practice there is a very sharp division between smeared packets and the non-smeared packets. Smeared packets have 26, 27 or 28 bytes in error while the others rarely have more than 10 bytes in error. We could also have defined a smeared packet to be one with close to half its bits in error (though for technical reasons this would be more complicated). Smeared packets were observed at 18 of the 35 sites. These sites are listed in Table 4.1 along with the number of smeared packets, the smeared packet rate (SPR), the quotient SPR/ABER and the bit error rate observed within smeared packets.

The ratios SPR/ABER are remarkably constant. There are outliers but on the whole the ratios are between 2 and 15. This suggests that the same noise phenomenon produces the random errors and the smeared packets by acting in different ways. We don't see half a smeared packet anywhere and the bit error rates are remarkably uniform within smeared packets. This suggests that the decoder is almost out of synchronization in these packets. It is possible that there are impulses of noise of duration less than 1/60 sec. which obscure one packet but leave the surrounding packets unaffected. But they wouldn't be likely to be so correlated to the ambient BER and one would expect bit error rates within the packets above and below 0.5.

What actually happens inside a smeared packet is a bit mysterious. Why is the bit error rate within smeared packets on average 0.467 with a small standard deviation? The bits sent are, after the first three bytes, a pseudo-noise

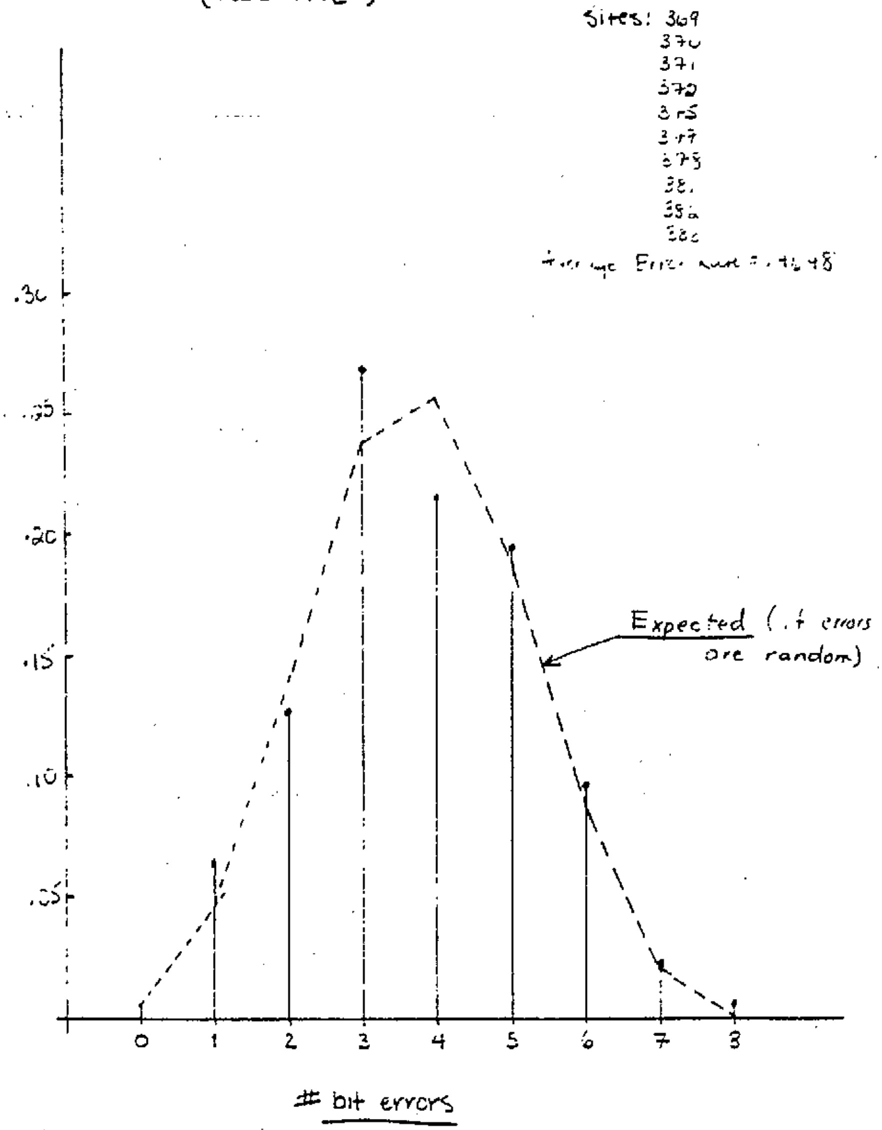
TABLE 4.1 SMEARED PACKETS.

Site Number	Location	Number of Smeared Packets	Smeared Packet Rate	SPR / ABER	BER inside smeared packets
362	Parkside	8	3.5e - 4	2.4	0.459
363	Wakaw	3	3.0e - 4	7.8	0.466
364	Crystal Springs	1	1.3e - 4	10.7	0.464
365	Birch Hills	10	9.8e - 4	9.5	0.451
367	Watrous	6	6.7e - 4	15.1	0.472
369	Dundern	7	7.7e - 4	24.5	0.460
370	Hanley	92	1.3e - 2	3.8	0.477
371	Delisle	6	7.5e - 4	64.0	0.461
372	Harris	18	6.4e - 4	7.4	0.474
375	Asquith	4	2.2e - 4	5.9	0.461
377	Maymont	3	2.3e - 4	3.5	0.490
378	Maymont	1	2.5e - 4	44.8	0.589
381	N. Battleford	25	8.6e - 2	26.0	0.408
382	Hwy. 29	2	1.1e - 4	.075	0.368
383	unknown	28	1.7e - 2	2.9	0.459
386	Kerrobert	64	2.8e - 2	5.5	0.480
387	Netherhill	60	3.0e - 3	3.5	0.481
388	Rosetown	51	2.0e - 3	5.7	0.485

sequence. Are these first three bytes more robust - enough to reduce the bit error rate from .5 to .467 when synchronization is lost?

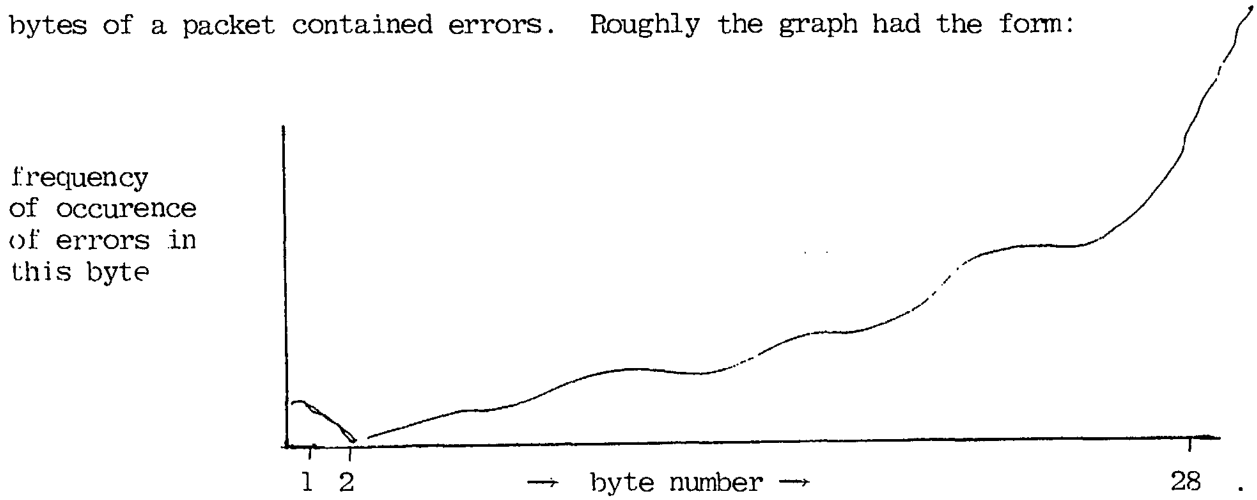
One further calculation was made. If the bits in a smeared packet are independently in error with a fixed probability p then we expect a byte with k errors with probability $\binom{8}{k} p^k (1-p)^{8-k}$, a binomial distribution. The average frequency of occurence of byte errors of weight k was calculated for 10 sites for $k = 0,1,\ldots,7,8$. This was compared to the frequencies expected under the binomial distribution with $p = .4648$, the average error rate in smeared packets at these sites. The results form Figure 4.5 We see that there is good agreement on the tails but in the central important part the two distributions don't agree. If the errors are not independent what are they? We don't know.

Figure 4.5 OBSERVED DISTRIBUTION OF BYTE ERRORS IN SMEARED PACKETS (RELATIVE)



4.4 The Bytes Within a Packet

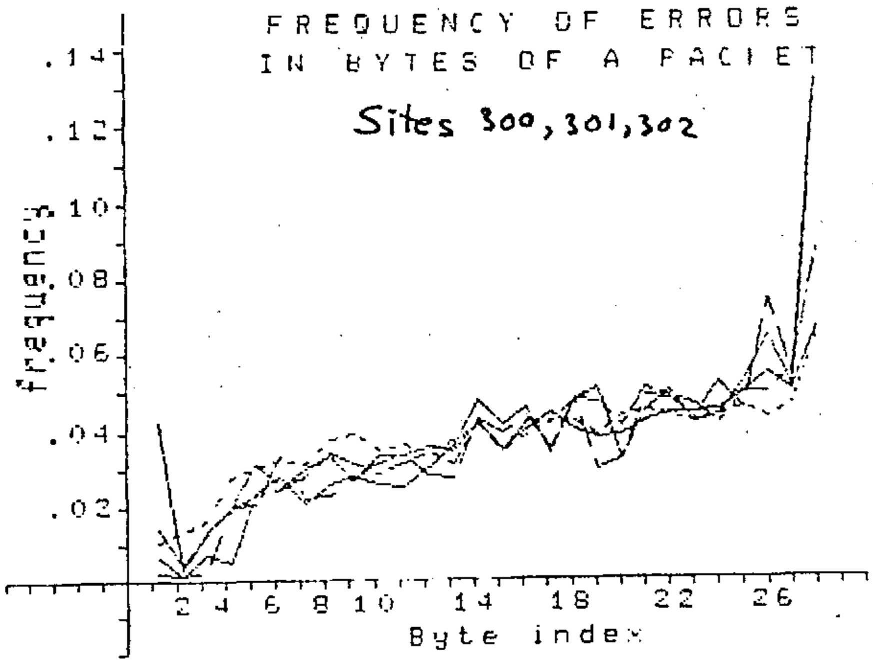
The earliest sites analysed were Sites 300, 301 and 302. At these sites it was observed that there was a definite trend in the frequency with which the 28 bytes of a packet contained errors. Roughly the graph had the form:

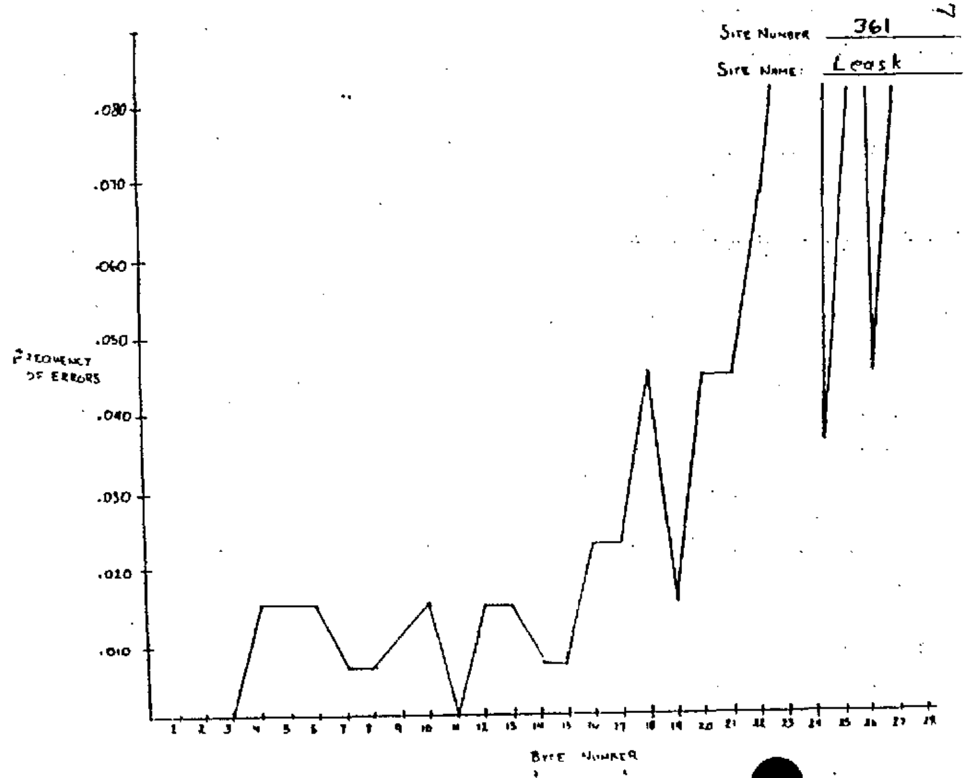


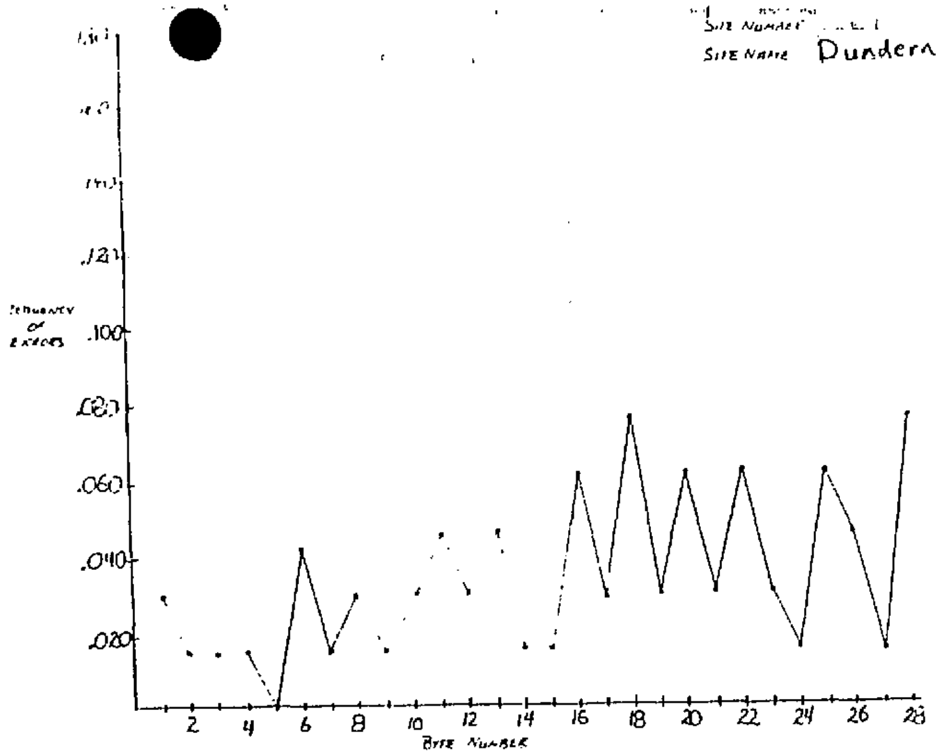
When we went on to the other 32 sites this was something for which we looked. We found an increasing likelihood of error in later bytes of a packet at Sites 300, 301, 302, 361, 372, 380, 382 and 385.

Byte index is plotted against frequency of errors in that byte for 23 sites in the accompanying graphs. A great variety of curves is obtained with the 25% of sites mentioned above being the only consistent sub-class.

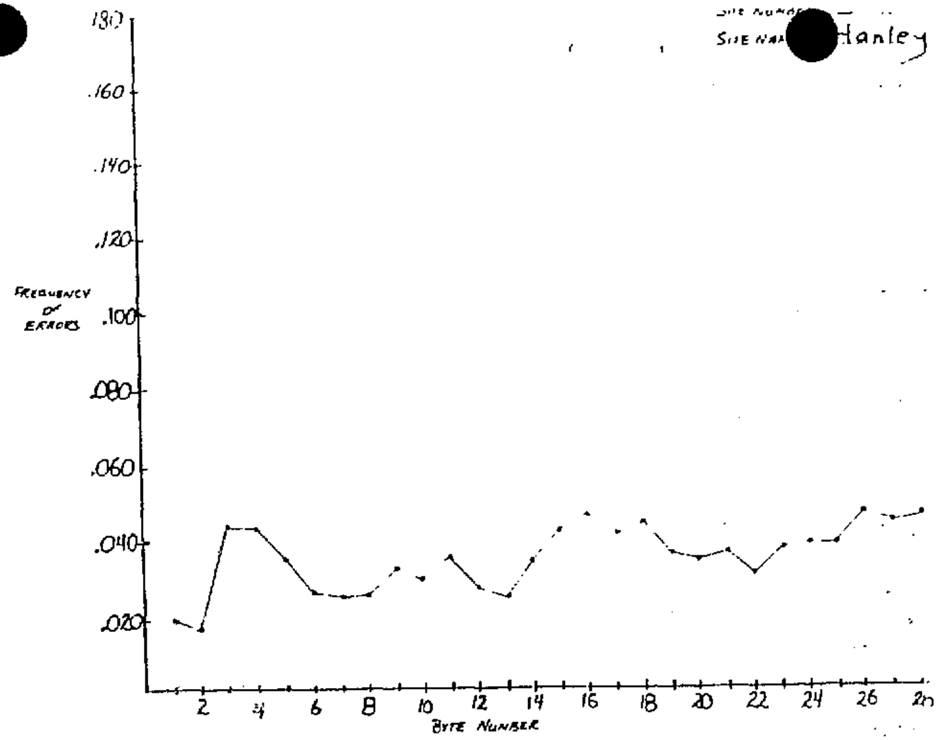
Figure 4.6 Frequency of Errors in Bytes of a Packet



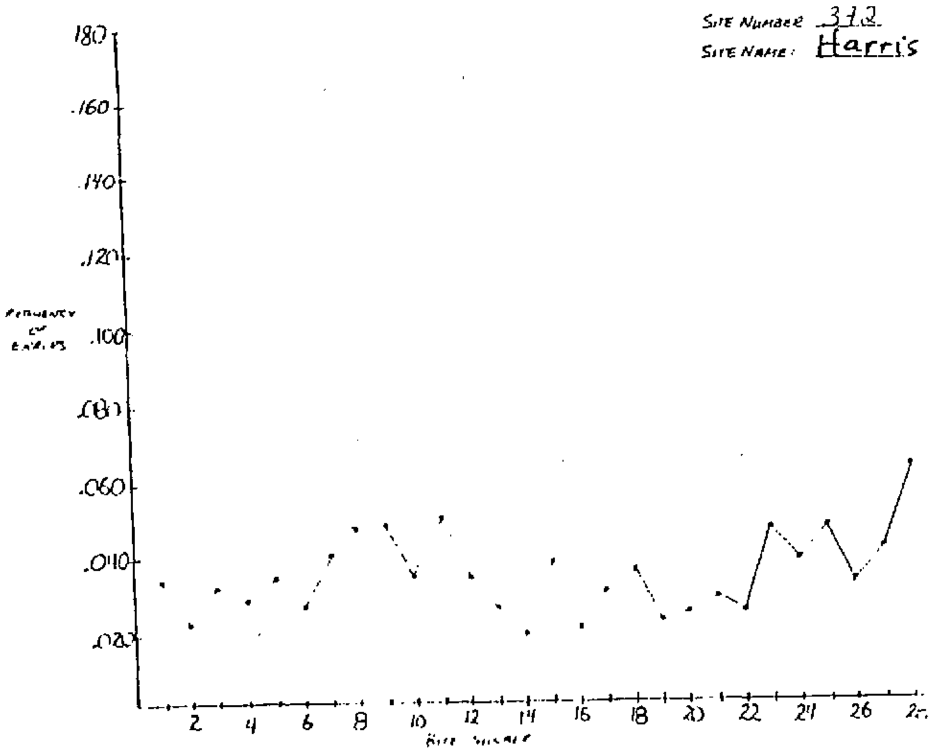




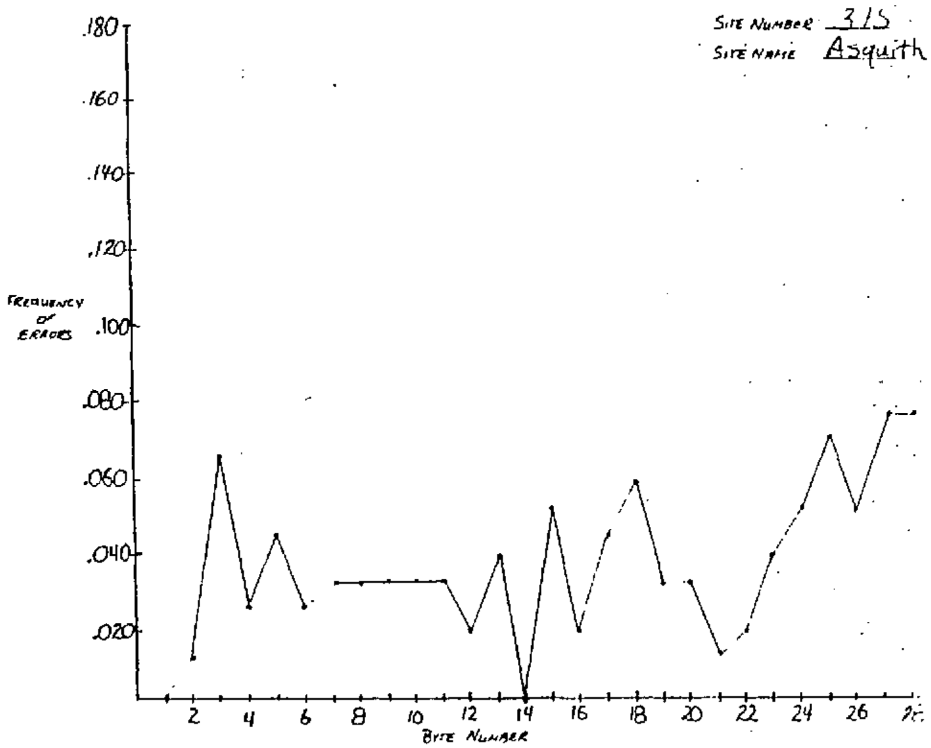
Site Name Dundera
Frequency of Errors
.100
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
Byte Number



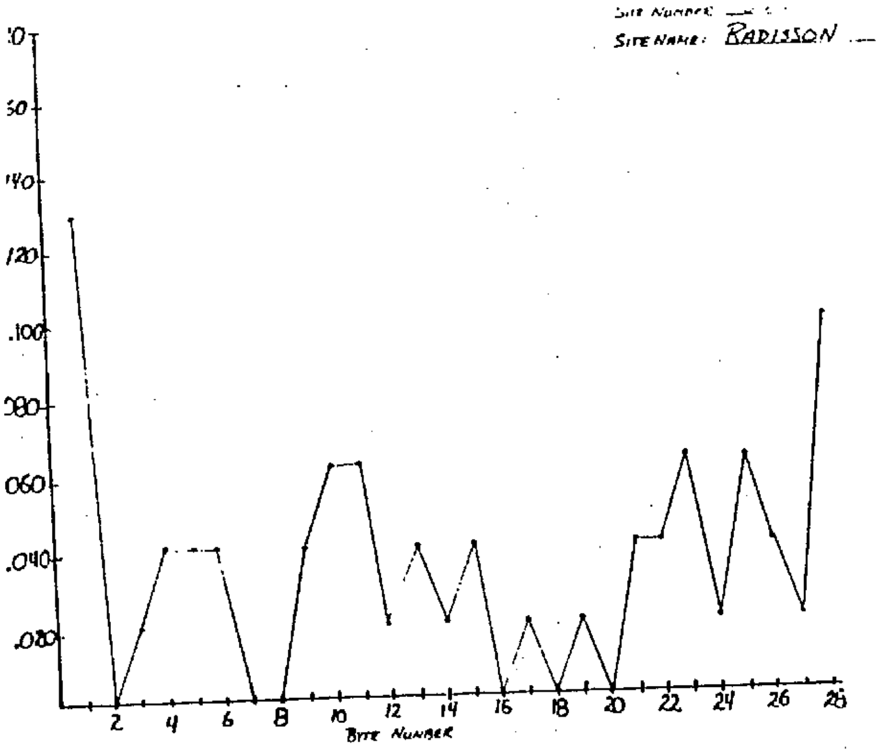
Site Name Hanley
Frequency of Errors
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26
Byte Number

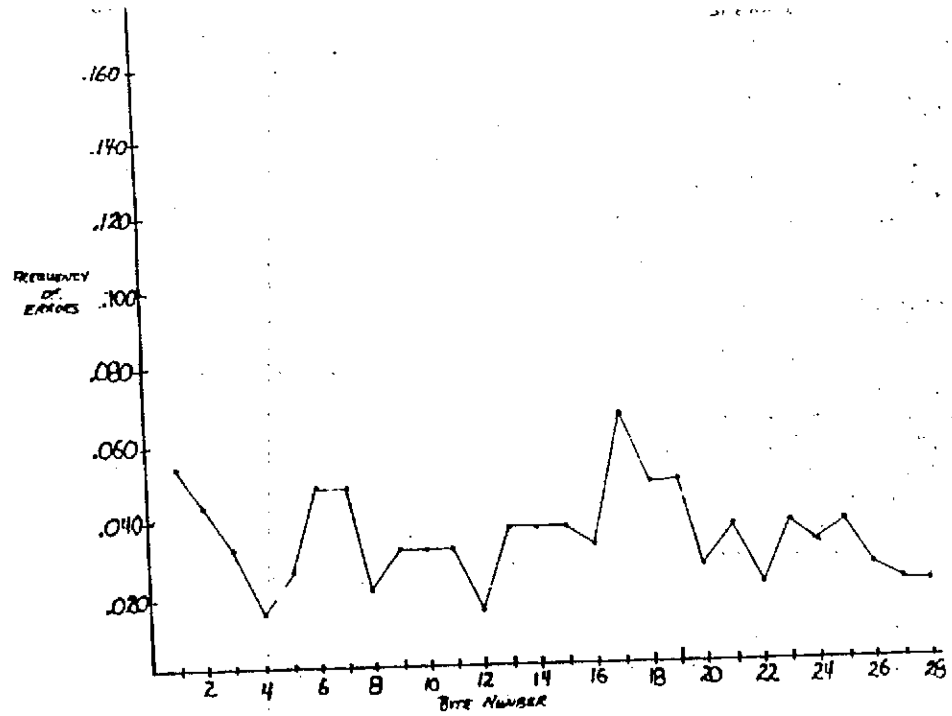


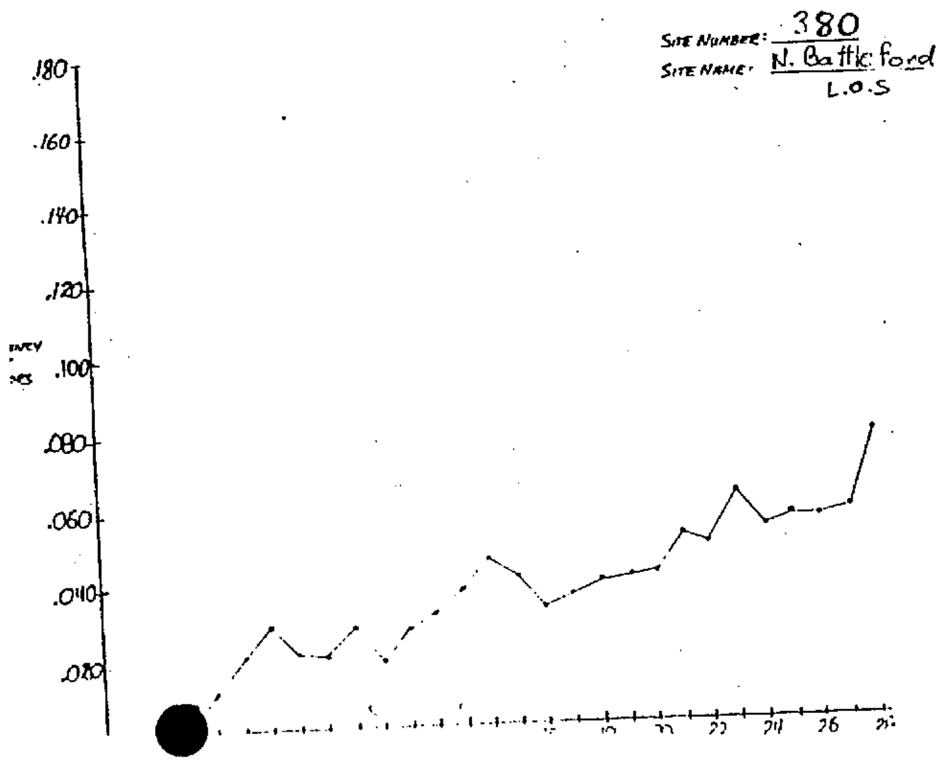
Site Number 312
Site Name: Harris
Frequency of Errors
180
.160
.140
.100
.060
2 4 6 8 10 12 14 16 18 20 22 24 26

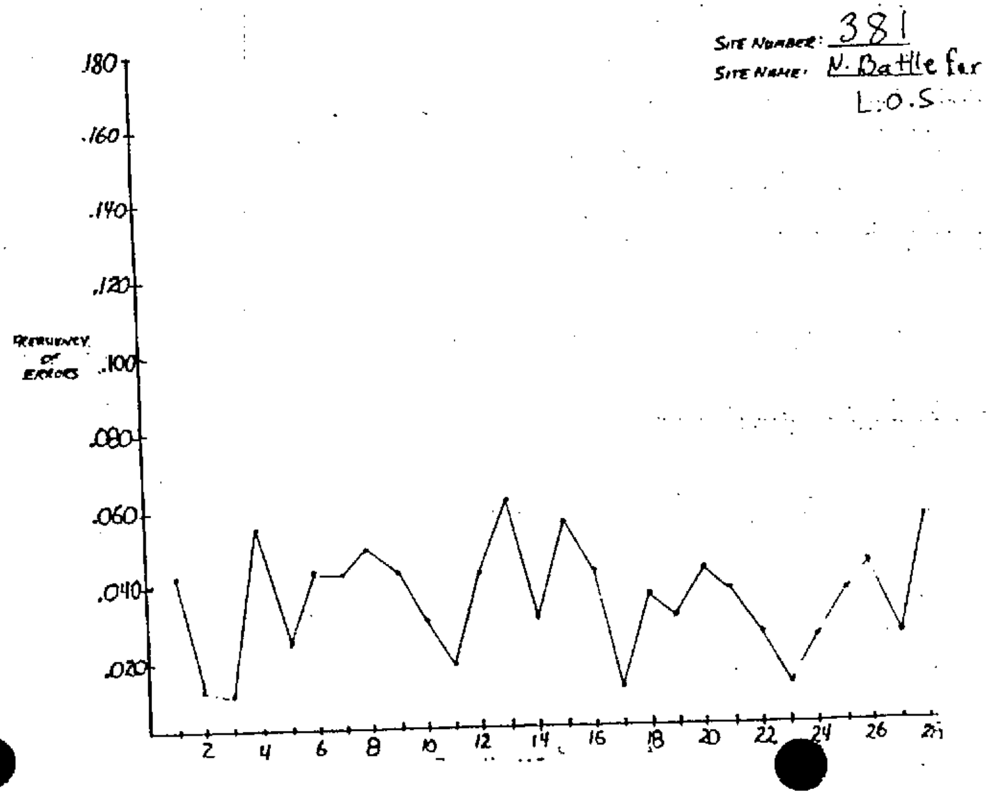


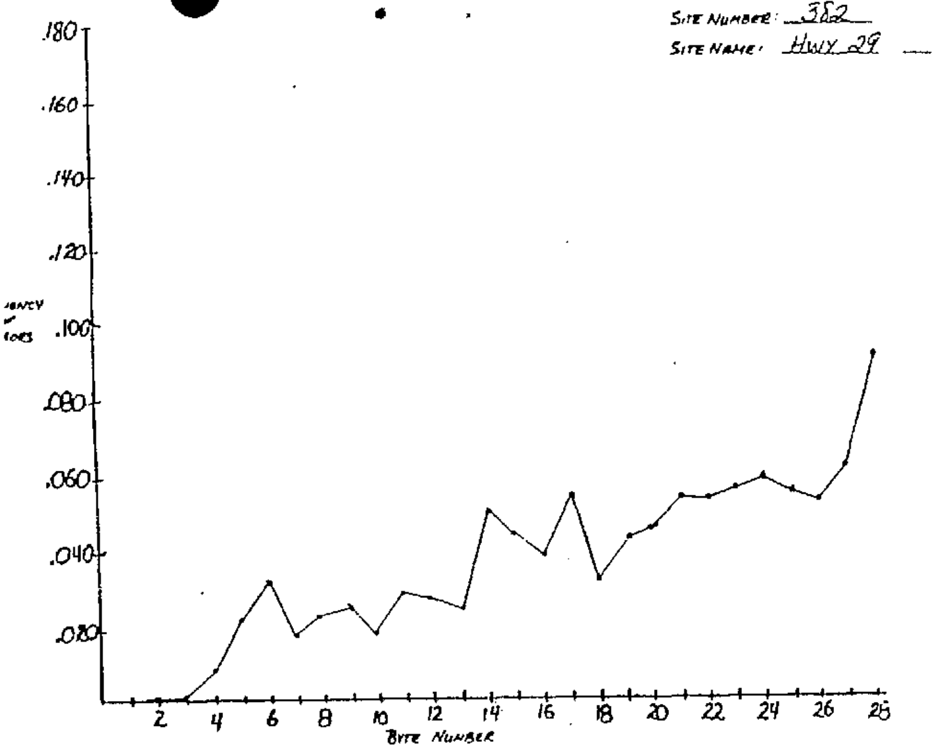
Site Number 315
Site Name Asquith
Frequency of Errors
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26
Byte Number

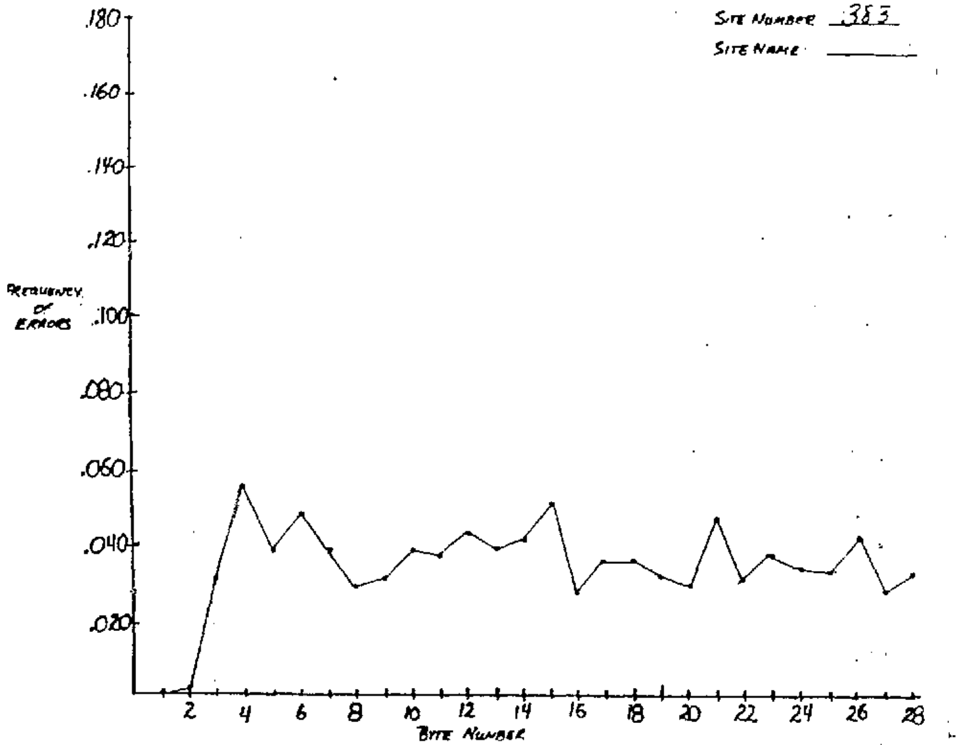

SITE NAME: RADISSON
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
BYTE NUMBER

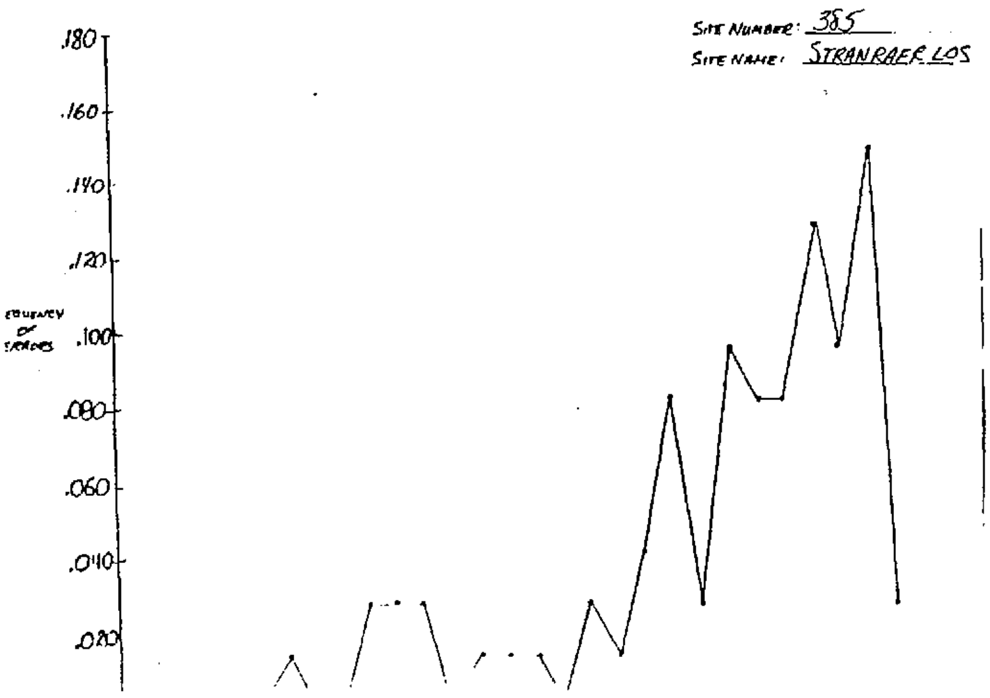

FREQUENCY OF ERRORS
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
BYTE NUMBER

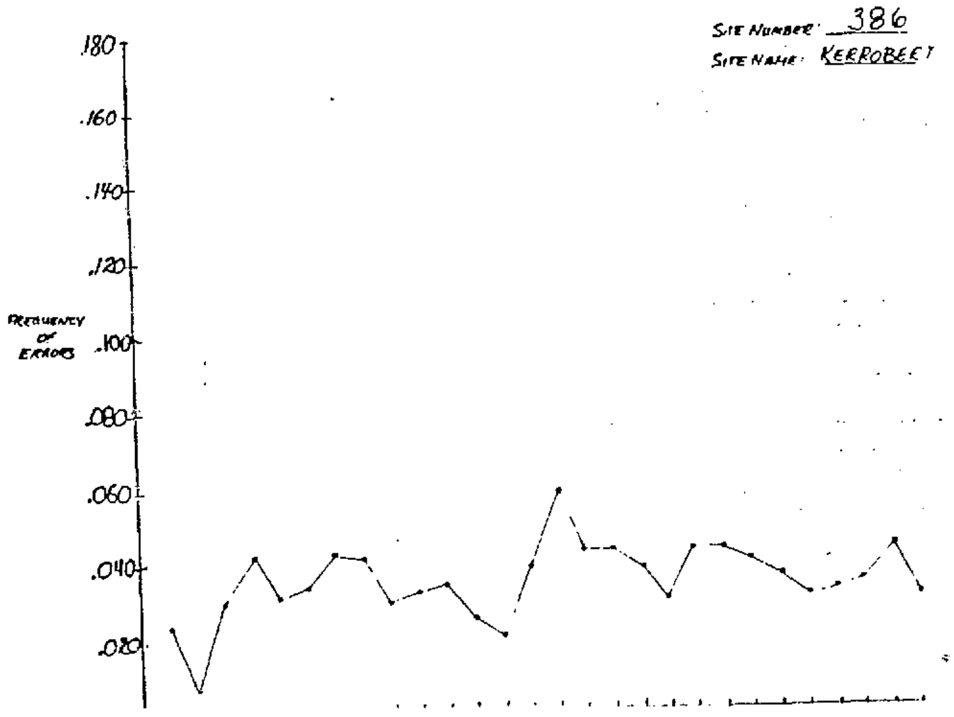

SITE NUMBER: 380
SITE NAME: N. Battleford
L.O.S
.180
.160
.140
.120
.100
.080
.060
.040
.020
20 22 24 26

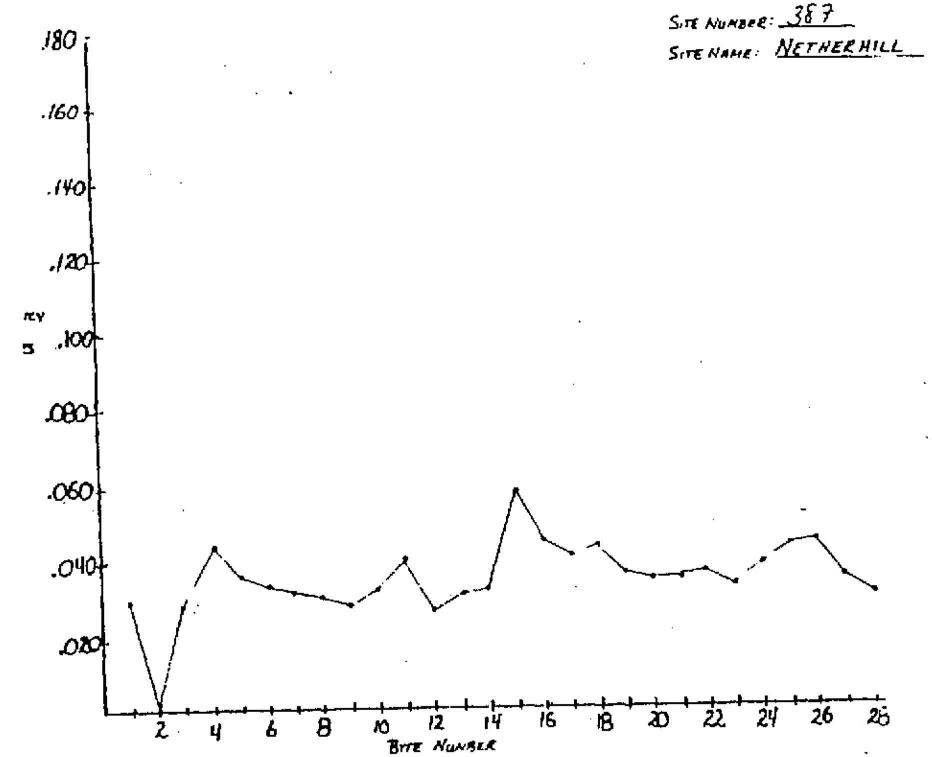

SITE NUMBER: 381
SITE NAME: N. Battlefor
L.O.S
FREQUENCY OF ERRORS
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26

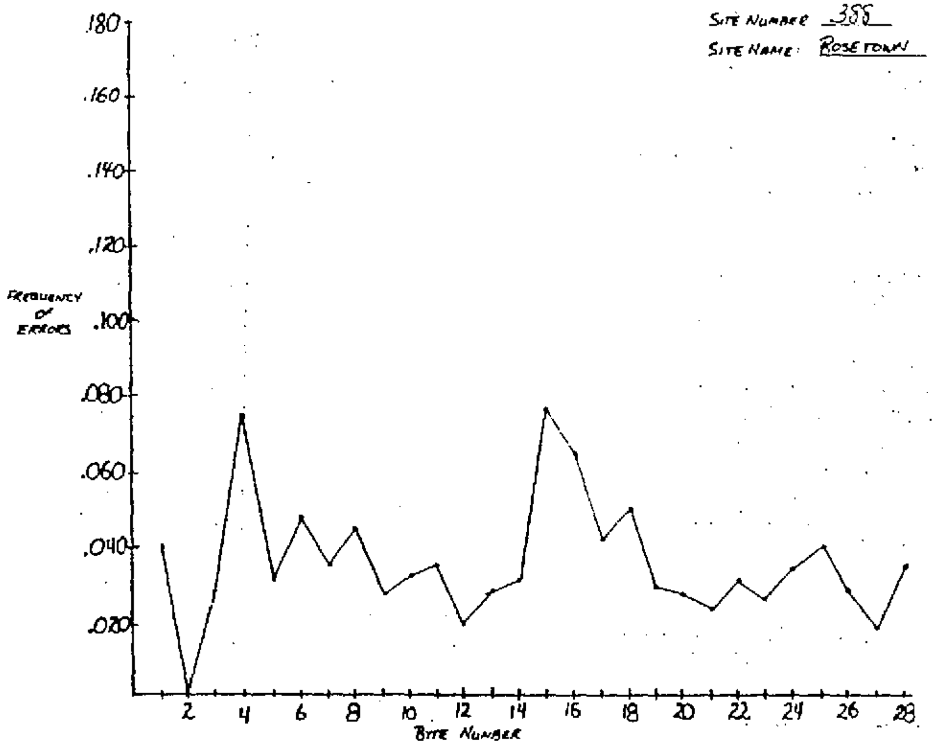

Site Number: 382
Site Name: HWY 29
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
Byte Number

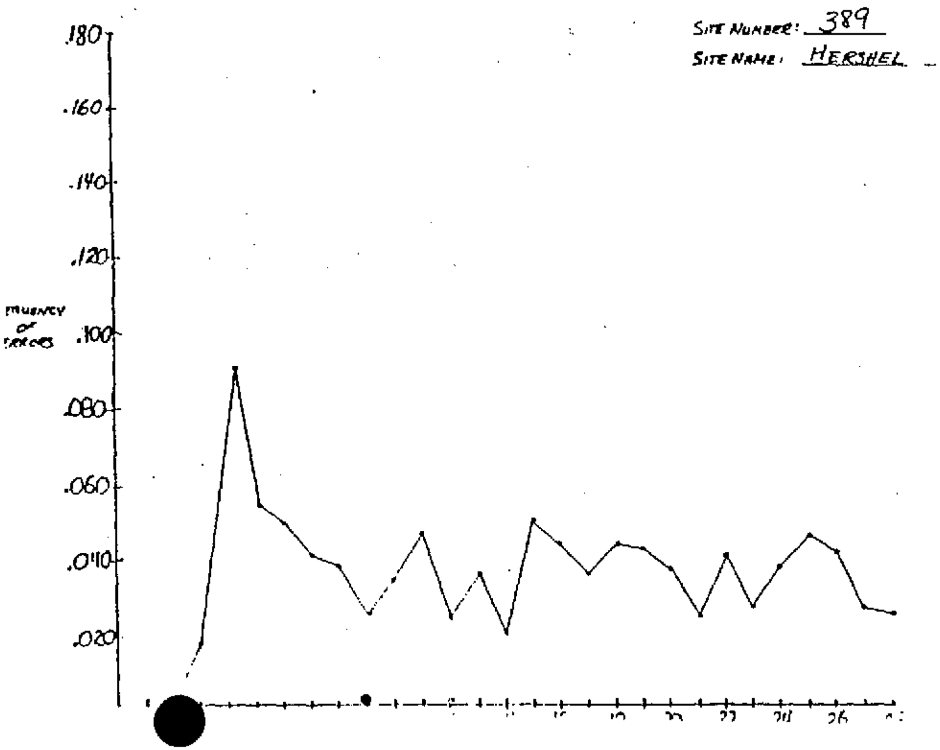

Site Number: 383
Site Name:
Frequency of Errors
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
Byte Number

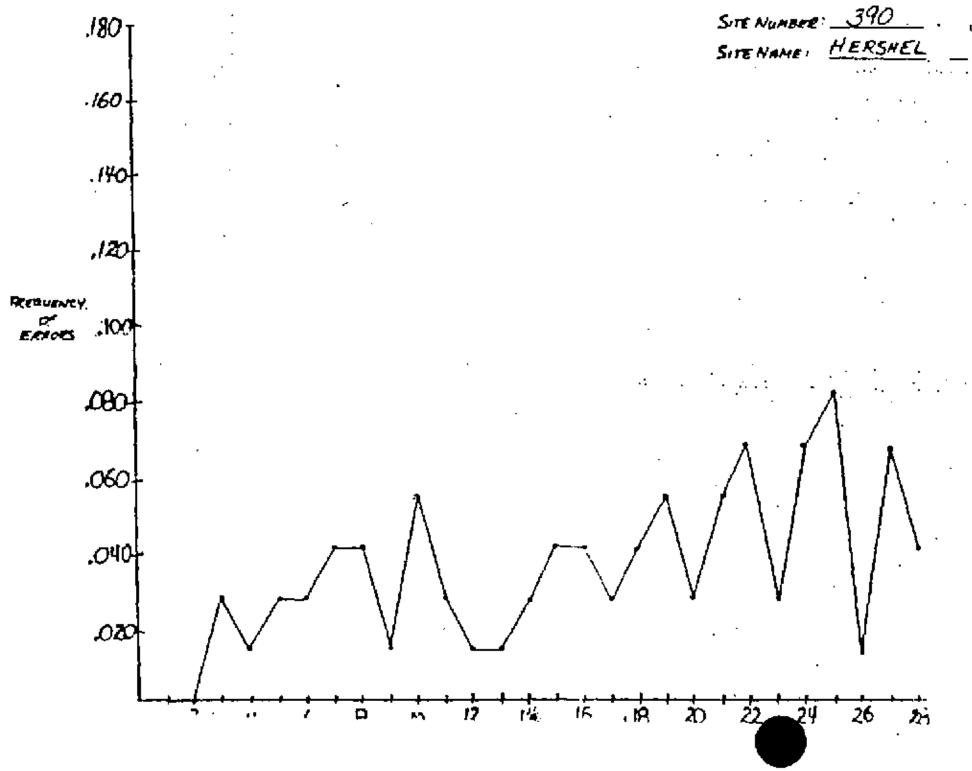

Site Number: 385
Site Name: STRANRAER LOS
.180
.160
.140
.120
.100
.080
.060
.040
.020

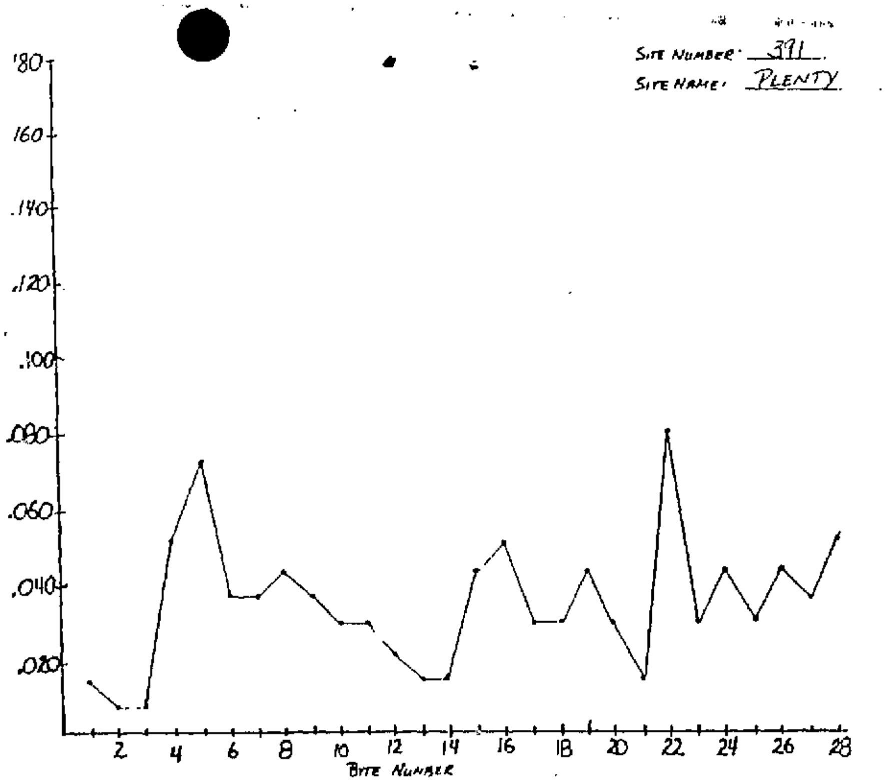

Site Number: 386
Site Name: KERROBEET
Frequency of Errors
.180
.160
.140
.120
.100
.080
.060
.040
.020

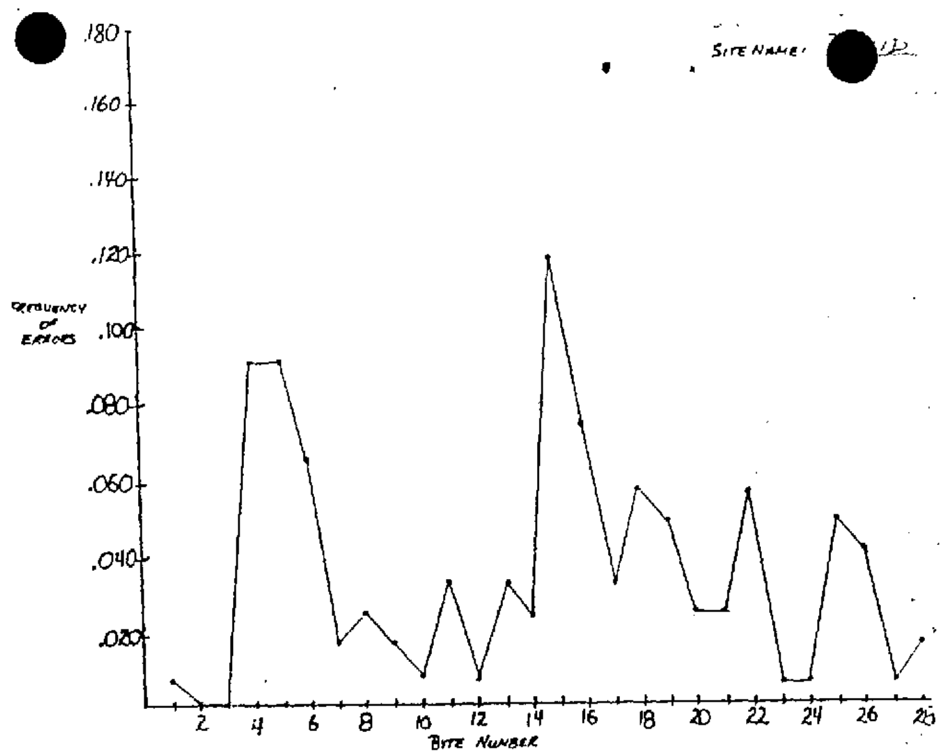

SITE NUMBER: 387
SITE NAME: NETHERHILL
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
BYTE NUMBER

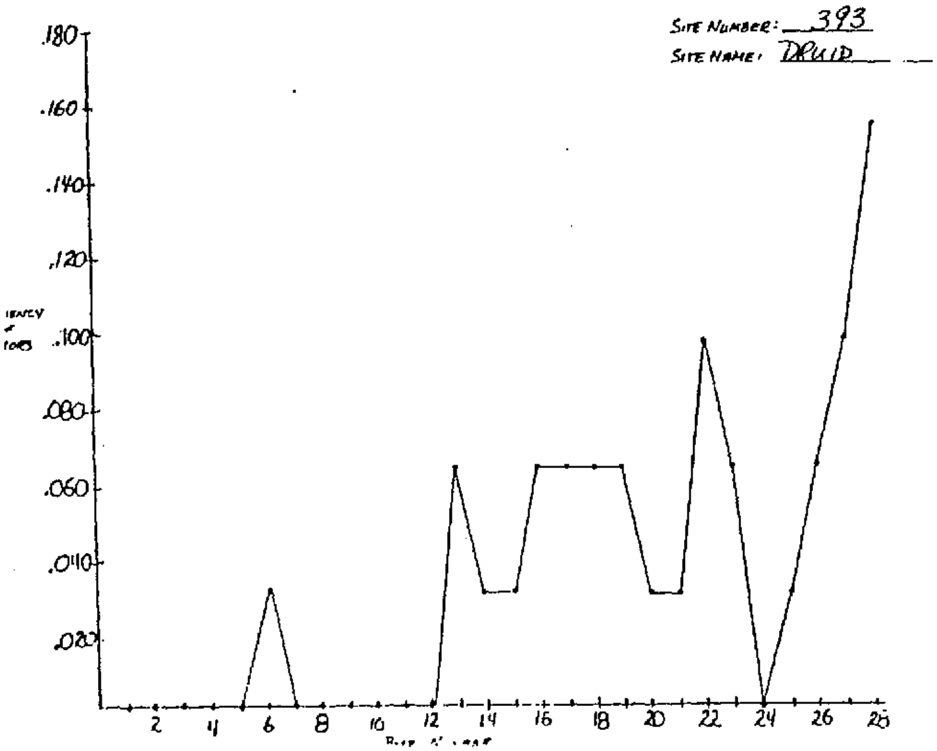

SITE NUMBER 388
SITE NAME: ROSETOWN
FREQUENCY OF ERRORS
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
BYTE NUMBER


SITE NUMBER: 389
SITE NAME: HERSHEL
.180
.160
.140
.120
.100
.080
.060
.040
.020


SITE NUMBER: 390
SITE NAME: HERSHEL
FREQUENCY OF ERRORS
.180
.160
.140
.120
.100
.080
.060
.040
.020


Site Number: 391
Site Name: PLENTY
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
Bite Number


Site Name:
.180
.160
.140
.120
Frequency of Errors
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28
Bite Number


Site Number: 393
Site Name: DRUID
.180
.160
.140
.120
.100
.080
.060
.040
.020
2 4 6 8 10 12 14 16 18 20 22 24 26 28

4.5 Bits in a Byte.

We isolated the bytes with single bit errors at each site and calculated the frequency with which each bit was in error. If errors are independent random events we would expect each bit to be the erroneous one close to 1/8 of the time. However at many sites this is not what was observed. For example consider Site 372:

bit number	:	1	2	3	4	5	6	7	8
number of times in error	:	61	60	58	44	56	50	139	53
frequency	:	.117	.115	.111	.085	.107	.096	.267	.102 .

Table 4.2 presents the frequencies at 27 sites where there are enough bytes with single errors to be significant. Boxed numbers are at least 0.200 which is 60% more than 1/8. Underlined results are at most 0.050 which is 60% less than 1/8.

We observe that bit 7 is badly jinxed with bit 8 also in trouble sometimes. The high frequencies in the bit 7 column are paid for uniformly by the other bits at most sites.

Table 4.2 Frequency of Error in the Bits of Bytes with One Bit Error.

Site #	1	2	3	4	5	6	7	8
300-1	.120	.101	.113	.146	.129	.093	.149	.148
300-2	.069	.116	.072	.238	.066	.110	.146	.182
301-1	.128	.127	.094	.136	.103	.122	.170	.119
301-2	.069	.149	.075	.168	.052	.146	.173	.168
302	.102	.118	.102	.151	.109	.100	.196	.121
361	.088	.074	.132	.132	.162	.118	.081	.213
362	.125	.141	.094	.090	.063	.086	.246	.156
363	.116	.023	.174	.093	.198	.116	.174	.105
364	*							
365	.091	.101	.130	.120	.096	.096	.216	.149
366	*							
367	.157	.090	.191	.112	.090	.090	.169	.101
368	*							
369	.094	.047	.109	.156	.094	.094	.219	.188
370	.105	.102	.108	.109	.106	.106	.241	.123
371	*							
372	.117	.115	.111	.085	.107	.096	.267	.102
375	.105	.125	.079	.105	.138	.099	.171	.178
376	.064	.085	.085	.043	.085	.106	.362	.170
377	.082	.110	.082	.143	.126	.099	.242	.115
378	*							
380	.123	.110	.099	.129	.130	.120	.142	.147
381	.086	.093	.093	.129	.129	.079	.236	.157
382	.128	.106	.093	.136	.132	.121	.135	.150

Site #	1	2	3	4	5	6	7	8
383	.098	.102	.123	.103	.121	.111	.194	.148
384	*							
385	.114	.071	.143	.086	.129	.171	.143	.143
386	.113	.108	.103	.111	.110	.110	.216	.130
387	.094	.098	.119	.097	.091	.095	.215	.191
388	.096	.101	.112	.101	.086	.094	.241	.170
389	.077	.089	.156	.094	.076	.063	.201	.244
390	.111	.056	.069	.220	.097	.111	.194	.139
391	.093	.039	.085	.093	.078	.101	.202	.310
392	.075	.067	.158	.150	.050	.025	.158	.317
393	*							

* : at these sites there are too few relevant bytes to be significant.

4.6 Effects Between Packets

As noted in Section 4.1, one would expect the errors in successive packets in the channel to be uncorrelated. We tested this in two ways. First the sequence of gaps between erroneous packets was considered as a time series. Thus after each erroneous packet there is a sequence of say g_i correct packets then an erroneous packet. The sequence of gap lengths g_i was considered.

$$--- g_{i-1}, g_i, g_{i+1}, ---$$

The autocorrelation function was calculated for this series for lags of up to 30 gaps. Secondly the distribution of gap lengths was calculated and compared with the distribution of gap lengths one would expect if erroneous packets are independent events.

In detail, the autocorrelation function of the gap sequence is calculated as follows. Let the sequence of gaps be $g_1, g_s, ---, g_i, ---, g_N$ and let $\bar{g}$ be the average gap length. Then the autocovariance coefficient at lag k is

$$C_k = \frac{1}{N} \sum_{t=1}^{N-k} (g_t - \bar{g})(g_{t+k} - \bar{g}) .$$

From this we compute the autocorrelation coefficients

$$r_k = C_k / C_o .$$

(See Chatfield [8] pg. 23-30.) These numbers give some indication of the relationship of adjacent or nearly adjacent gap lengths. If the gap lengths are randomly distributed then the autocorrelation coefficients will be approximately distributed as a normal variable with mean 0 and variance $\frac{1}{N}$ where N is the number of gaps. This then gives us a test for random distribution of the gap lengths. Following Kendall and Stuart ([9], Chapter 48) we declare a coefficient r_k to be significant if it lies outside the bounds $-\frac{1}{N} \pm \frac{2}{\sqrt{N}}$. (95% confidence limits). A small number (≤ 4) of significant coefficients would be expected even if the gap lengths appear randomly. There were however 8 Sites (300-2, 301-1, 302, 377, 380, 382, 387, 388) where a large number of coefficients were significant.

At Sites 377, 387 and 388 the noise level in the channel fluctuates enough to make the gap lengths in part of the series all greater than $\bar{g}$ and in other parts all less than $\bar{g}$. Thus it is a varying channel which produces the effect. At Sites 302, 380 and 382 there is a great deal of noise and an unexpectedly large number of gap lengths are 0. The result is that gap lengths are correlated to nearby gap lengths in a positive way. At Sites 300-2 and 301-1 there is roughly the expected number of gaps of length i for $i \leq 30$ and still a positive correlation in the gap lengths. Therefore at 5 sites we observed definite correlations of packet errors and at 30 sites no such correlations.

The second test looked at the number of times a gap of length k was observed. If the probability of an erroneous packet is p then one expects to observe $p \cdot (1-p)^k N$ such gaps if the gaps are independent. Gap lengths up to 30 were considered. The Kolmogorov-Smirnov statistic was calculated to decide whether there was a significant deviation from the expected number of occurrences of gaps of length k. At a 5% confidence level, 14 sites showed significant deviations from the expected distribution of gap lengths. Note though that we are really excluding sites with a large average gap length since then for $k \leq 30$ both expected and observed numbers of gaps of length k are zero. These sites are those with say $\bar{g} > 100$ which means 14 sites with no significant deviations for this reason. So effectively 14 of 21 or 66% of the sites considered had significant deviations from the expected number of gaps of length k for $k \leq 30$. We haven't done enough detailed study to make strong conclusions here but there appears to be an excess of short gap lengths at many sites but from our earlier test, there are many of these sites where the gap lengths are still randomly shuffled.

REFERENCES

[1] Brian Mortimer and Michael Moore (1984),Error Correction Schemes for Broadcast Teletext Systems, DSS Contract No.OST83-00078, Final Report.

[2] Brian Mortimer (1983),Error Correction Schemes for Broadcast Teletext Systems, DSS Contract No.OST83-00078, Progress Report.

[3] Brian Mortimer and Michael Moore (1983),More Powerful Error-correction Schemes for the Broadcast Telidon System, DSS Contract No.OSU82-00164, Final Report.

[4] Brian Mortimer and Mike Moore (1982),Two Byte Data-Block and Bundle Codes for the Broadcast Telidon system ,DSS Contract NO.OSU82-00164, Progress Report.

[5] Brian Mortimer (1982),Error-correcting Codes for the Canadian Broadcast Telidon System, DSS Contract NO. OSU81-00095, Final Report.

[6] O.Yamada, "Development of Error-Correcting Method for Broadcast Data Packets Multiplexed on NTSC TV Signals", source and date unavailable.

[7] "Simmulation of Error Correction for Japanese Teletext", CCIR(1982-1986), Study Groups Document 11/29E(Japan), 6 May, 1983.

[8] C. Chatfield, The Analysis of Time Series: Theory and Practice, Chapman and Hall, London, 1975.

[9] M.G.Kendall and A.Stuart, The Advanced Theory of Statistics, Vol.3, Griffin, London, 1966.