P2.1 Lossless Coding and Compression of Radar Reflectivity Data

V Lakshmanan
National Severe Storms Laboratory & University of Oklahoma [*]

Abstract

The coding and compression of weather imagery can be improved by following a domain-specific approach in which the characteristics of the data are taken into account in devising the encoding scheme. We show that by predicting that the next valid data element is identical to the current one, a dramatic reduction in source entropy (information content) is achieved. We also show that encoding runs of missing values in combination with such a prediction scheme makes further compression unnecessary. Finally, the method introduced in this paper is compared with existing methods and shown to outperform all of them.

1. Introduction

A study was carried out on seventy volumes of radar reflectivity data collected by the Weather Service Radar in Fort Worth, Texas on May 5, 1995. These volumes provide an indication of the distribution of reflectivity data on a typical storm day. Since the amount of valid data is higher on a storm case, these volumes also serve as a worst case scenario from a compression stand-point. In the storm cases we considered, even considering only the lowest tilt of the radar where the highest portion of valid data values are to be found, nearly 60% of the data elements in the case were missing. The distribution of the various data values in the lowest elevation tilts of the radar over the entire case is shown in Figure 1a.

It may be noted from Figure 1a that there is a spread of reflectivity values over which the data are distributed. The information content in the case may be obtained from the Shannon entropy (Shannon, 1948), defined as: $h = -\sum_i p_i log_2 p_i$ where the summation is over all the possible symbols[1] in the data set. Defining the entropy this way makes the implicit assumption that the symbols we are summing over are all statistically independent. Because radar data tends to be correlated, entropy calculated in this manner can be misleading. If the data are decorrelated, then the entropy equation may be used to compute the true information content.

[*]lakshman@nssl.noaa.gov

[1]The symbols are what needs to be represented. Each symbol could correspond to an actual data value, e.g. $3\ dBZ$, or to a value in some transformation of the data.

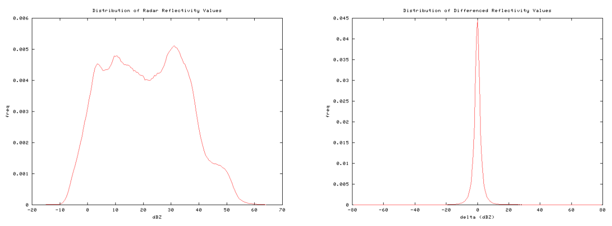

Figure 1: (a) Distribution of radar reflectivity values in $-15\ dBZ$ to $64\ dBZ$ in $0.5\ dBZ$ increments in the lowest elevation tilt of the radar. Missing values are not shown in this figure; if they had, they would have been at 0.59. (b) Distribution of the error in the predicted valid reflectivity values. Missing values are considered a separate class and are not included in the prediction algorithm.

Because Level II reflectivity data ranges from $-15\ dBZ$ to $64\ dBZ$, in increments of $0.5\ dBZ$, the number of symbols corresponding to valid data is 159. With missing data considered another symbol, we have 160 symbols over which the entropy may be calculated. For the data case studied, whose distribution is shown in Figure 1a, the entropy was 3.74 bits per symbol. Thus, a typical elevation tilt that contains 460 gates and 365 radials of reflectivity data (167900 symbols) can be represented using less than 80,000 bytes[2] with an ideal encoding.

2. Methods

Reflectivity data tends to be highly correlated. We can use this correlation between successive data elements profitably by dividing the data into two classes – valid reflectivity values and missing data – assigning a symbol to missing data directly, and doing a linear prediction on valid reflectivity values. We are “predicting” that the next range gate with valid reflectivity data has the same reflectivity as the current valid data value.

Doing the linear prediction (LP) this way has several advantages. By keeping the two classes separate, we re-

[2]167900 symbols $\times$ 3.74 bits/symbol $\times$ 8 bits/byte

Elevation	Raw	RLE	RLE+LZ	LP	LP+BSH	LP-RLM	LP-RLM+BSH
0.5	169280	146020	61167	66497	47998	43456	43660
2.4	131008	58214	27263	35879	21900	19078	19460
3.3	98624	40904	19198	26121	15629	13272	13679
4.3	79488	30862	14346	20186	11824	9847	10281
5.2	79272	26306	12170	18610	10004	8373	8794
6.2	64592	23014	10552	15455	8668	6919	7375
7.5	50279	20184	9723	13247	8201	6958	7247
8.7	46609	18062	8728	11962	7332	6227	6456
10.0	40370	16294	7931	10509	6612	5693	5893
12.0	36600	14292	6910	9252	5767	5016	5160
14.0	32850	12810	6221	8127	5070	4474	4597
16.7	29120	11394	5582	7068	4481	3971	4029
19.5	25340	10202	4933	6241	4012	3618	3628

Table 1: A comparision of various lossless compression methods on a volume of radar reflectivity data. RLE is run-length encoding; LZ is Lempel-Ziv compression; LP is linear prediction of valid data values; BSH is block-sorting followed by Huffman coding; LP-RLM is linear prediction of valid data values combined with run length encoding of missing values. The data corresponds to a supercell thunderstorm in Fort Worth, Texas in March 2000. LP-RLM significantly outperforms all other methods and is not further compressible.

duce the dynamic range of our symbol set since straightforward numerical difference will incur jumps whenever the data change over from missing to valid or valid to missing. Because of the frequency of missing data, we will allocate a single bit to it. Thus, linear prediction of missing data can not decrease the bit entropy of the encoded data. Computing the difference, rather than doing run-length encoding, allows us to profitably use not only a zero-change between successive elements, but also small changes. In fact, for the same data case whose reflectivity values are shown in Figure 1a, the distribution of the various symbols (except the missing ones) is shown in Figure 1b. The symbols correspond to a range of $-79\ dBZ$ to $79\ dBZ$. The Shannon entropy for the data, after being processed in this manner, is only 2.82 bits per symbol.

Huffman (1952) derived a method of assigning bits to symbols where more frequent symbols receive shorter bit sequences in such a way that the entropy of the encoding scheme is within 1% of the Shannon entropy, the theoretical limit. Huffman's encoding method was used to obtain a way of representing the linear-predicted stream of symbols. The encoding had a bit entropy of 2.85 bits/symbol, very close to the source entropy of 2.82. The encoding, by taking care to assign shorter sequences to more frequently occuring data elements, in itself, leads to a Huffman code with a bit entropy of about 2.85 bits/symbol. Thus, a typical elevation tilt that contains 460 gates and 365 radials of reflectivity data (167900 symbols) can be represented using less than 60,000 bytes in a storm environment. This is an improvement of 25% over directly encoding the data.

In clear air, we would require one bit for each missing data element and can therefore represent the data using just 20,000 bytes. Yet, there is not 20,000 bytes of information in a clear-air tilt. Although the encoding is very efficient, we can compress the data further by transforming it further. While we have arranged to assign short bit sequences for commonly occuring data elements, we still have not used the fact that certain strings of symbols may repeat within the data.

We considered two methods of transforming the Huffman encoded stream of linearly predicted symbols in order to decorrelate the data: (a) Compress the linear predicted data using a general purpose compression technique. This works because general purpose compression techniques try to achieve an output where the output bits are statistically uncorrelated. If the output bits were correlated, then the compression technique could have done better. (b) Use run-length encoding on the missing symbols. The implicit assumption is that the predicted errors in the valid values are uncorrelated, i.e. that the reflectivity data are a first-order Markov process with an autoregressive coefficient of 1. In this case, run-length encoding the predicted errors in the valid values would not have much of a pay off.

It is possible to apply a reversible transformation to the data so that these strings of symbols repeat (Burrows and Wheeler, 1994). The reordered set of strings is then Huffman coded once again, this time using an encoding scheme based on the reordered data. Since the Huffman coding in this case changes from tilt to tilt (depending on the data), the coding is stored with the data.

The results here vary, but in the worst case, a tilt of 460 gates and 365 radials containing storm data can be represented using about 48000 bytes, an improvement of about 20% over the linear predicted encoding scheme. When there is less valid data, the size of the encoded data can drop to about 4000 bytes.

Having recognized that it is runs of missing data that are compressible further, we can choose to run-length encode missing data alone. The process, then, is as follows during the compression stage: 1. Initialize the predictor and the missing counter to zero. 2. If the next value is missing, increment missing counter. 3. If the next value is valid, find Huffman code for the difference from prediction. Write code for missing, followed by run length. Then, encode the valid difference. Reset missing counter to zero and the predictor to current valid value. This is a reversible transformation.

We can formulate a Huffman code for the distribution of predictor errors, missing values and missing run lengths. If the run length is more than the number of available symbols, then the data is simply encoded as several shorter runs.

The source entropy of this process turns out to be about 4.78 bits per symbol with a Huffman representation of 4.82 bits per symbol. Although the entropy seems to have increased, the reader should note that because of the run-length encoding process, the number of symbols has decreased. On both the training and the test cases, this method proved to be more efficient than simply using linear prediction on the valid values.

Following such an encoding with block-sorting compression and an adaptive Huffman code (BSH) is counter-productive. The size of the data increases due to the need to store the adaptive code.

3. Results and Discussion

Results are presented for a complete volume of radar reflectivity collected at Fort Worth, Texas on March 28, 2000, a test case that is different from the case that was used to obtain the distributions of reflectivity values shown in Figure 1. The data are encoded and compressed using various schemes currently used for radar data, as well as more general purpose methods available. The details are shown in Table 1.

The method of encoding used operationally in Weather Service Radar (WSR-88D) products is run-length encoding (RLE) (OSF, 1998). As can be noted from Table 1, the method described in this paper (LP-RLM) outperforms RLE by nearly a factor of three. Following the run-length encoded sequence by a good general purpose compression algorithm such as the Lempel-Ziv method (Gailly, 2000; Ziv and Lempel, 1977) reduces the size of the data significantly. Again, from the table, it is seen that the domain-specific linear prediction encoding scheme described in this paper achieves sizes close to the RLE+LZ version, at least in those tilts with significant amounts of data. Following the linear prediction with a block sorting compression algorithm (LP+BSH) improves compression such that this method outperforms RLE+LZ by about 20%. Using a combination of linear prediction on valid values along with a run-length for missing ones (LP-RLM), and using a combined Huffman code improves the result of even LP+BSH significantly. Compressing the result of the LP+RLM method with block sort actually increases the size, showing that the result of the LP+RLM encoding is not further compressible.

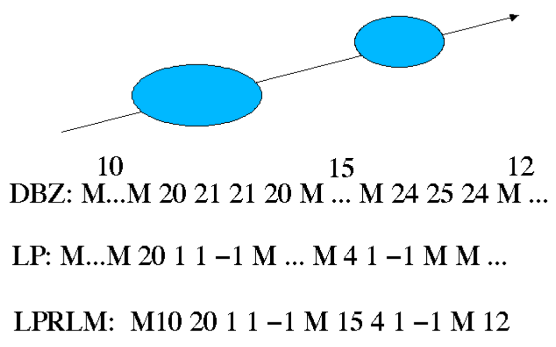
10 15 12
DBZ: M...M 20 21 21 20 M ... M 24 25 24 M ...
LP: M...M 20 1 1 −1 M ... M 4 1 −1 M M ...
LPRLM: M10 20 1 1 −1 M 15 4 1 −1 M 12

Figure 2: Encoding a radial of reflectivity data with the values given in the first row, i.e. 10 missing values followed by $20\ dBZ$, $21\ dBZ$, etc. LPis Linear Prediction while in LPRLM is Linear Prediction with run-length encoding of missing values.

References

Burrows, M. and D. Wheeler: 1994, A block-sorting lossless data compression algorithm. Technical Report 124, Digital Equipment Corporation, Palo Alto, California, available via http from gatekeeper.dec.com/pub/DEC/SRC/research-reports/abstracts/src-rr-124.html.

Gailly, J.-L.: 2000, Gzip. http://www.gzip.org.

Huffman, D.: 1952, A method for the construction of minimum redundancy codes. *Proceedings of the IRE*, **40**, 1098–1101.

OSF: 1998, Interface control document for product specification. Technical Report 2620003A, WSR-88D Operational Support Facility, Norman OK, available on request from OSF 3200 Marshall Ave, Ste 100, Norman OK 73072.

Shannon, C.: 1948, A mathematical theory of communication. *Bell System Tech. J.*, **27**, 379–423.

Ziv, J. and A. Lempel: 1977, A universal algorithm for sequential data compression. *IEEE Trans. on Information Theory*, **IT-23**, 337–343.