Wavelet compression for floating point data – Sengcom

Steve Sullivan
UCAR
University Corporation for Atmospheric Research
www.ucar.edu

October 11, 2012

Abstract

Sengcom (short for scientific and engineering data compression) is a new wavelet compression system for for scientific and engineering floating point data in general, and NetCDF4 and HDF5 floating point data in particular. Sengcom's goal is to provide better compression than the existing compression choices for NetCDF4 and HDF5. Sengcom gives the user tight control over the trade off between better compression and the maximum absolute error of the reconstructed values. Sengcom addresses 32 bit and 64 bit floating point data, not integer data.

This paper reviews many possible approaches and presents both the plan and performance results for a prototype implementation.

FEDERAL AVIATION ADMINISTRATION

Contents

1 Introduction

Meteorological data is of critical importance to the aviation community in planning flights, in pilot decisions, airport management decisions, scheduling, and other areas. In addition to the aviation community, meteorological data is important in operational decisions in the power generation and distribution, shipping and trucking industries, agriculture, and other sectors.

This paper discusses compression for floating point scientific data in the NetCDF4 format [45]. NetCDF4 is a file format based on the HDF5 format [19]. In essence, NetCDF4 is a subset of HDF5. Both NetCDF4 and HDF5 are widely used for the storage and transmission of meteorological, geophysical, and general scientific and engineering data.

NetCDF currently gets about 85,000 downloads per year, plus about 12,000 for the NetCDF Java version [46]. Currently Unidata's THREDDS data service provides about 1 TB / month of data in NetCDF format. Additionally, many recent and upcoming NSF meteorological studies require the data to be available in NetCDF format.

1.1 The need for a new compression system

As meteorological models and observations increase in frequency and resolution, the quantity of output data increases dramatically. This impacts

- The cost of storage. Despite the decreasing costs of disk storage, the overall costs seem to be increasing because the data volume is increasing quickly.
- The cost of transmission. Distributing large datasets to disparate locations requires expensive bandwidth provisions.
- The time of transmission. Retrieving a large dataset or distributing it to many sites can be time consuming.

2 Prior work

The history of scientific data compression and the associated fields of image and video compression is long and rich. Since bandwidth and storage capacity cost money, there have been strong economic incentives to find better methods for image, video, and scientific data compression.

A summary of our review of existing systems is in appendix A. Unfortunately **not one** of the existing systems meets our requirements, which are described in section 3. The most frequent reasons that existing systems fail to meet our needs are the following.

- The system must provide lossy compression - see the discussion in section 3.
- The system must have no intellectual property constraints on any use, even commercial use.
- The system must handle 32 and 64 bit floating point data.

Some existing compression systems deserve special note.

- Grib2 [56][57] uses a variety of compression systems. The most prevalent is JPEG2000, discussed below.
- Lempel-Ziv, or LZ systems [61] are based on the work of Ziv and Lempel in 1977. They are the foundation of zlib, zip, gzip, and similar systems. LZ systems are lossless - see the discussion in section 3.
- JPEG2000 [21][2] is a wavelet based image compression system that can be used either in lossy or lossless mode. It has a huge number of features related to image handling, such as region of interest handling. Although some JPEG2000 documents claim the system handles 32 bits, the implementations reviewed for this paper handle at most 16 bits.
- The SZIP [18] compression system is an optional compression module for HDF5. There are two reasons SZIP does not meet our needs.
  1. It is lossless. See the discussion in section 3.
  2. It has restrictive licensing. The SZIP license page reads in part:

     > Commercial use Commercial users may use the Szip software integrated with HDF to decode data and for internal activities that do not involve or result in the development of an Szip-based software product.
     > To use this software to encode data or in the development of an Szip-based software product, a commercial user may have to acquire an appropriate license from the appropriate licensing agent. See the HDF/Szip collaborative agreement (PDF) for details.
     > SZIP embodies certain inventions patented by the National Aeronautics & Space Administration. United States Patent Nos. 5,448,642, 5,687,255, and 5,822,457 have been licensed to ICs, LLC, for distribution with the HDF data storage and retrieval file format and software library products.
     > All rights reserved.
- FPC [7] is a compression system for 8 byte floats written by M. Burtscher and P. Ratanaworabhan at Cornell U. It is lossless so suffers the same problems as SZIP above. Also FPC is licensed for academic use only, making it not usable within NetCDF and HDF5.

Appendix A covers the remaining systems we reviewed.

3 Criteria

3.1 Data precision metrics

3.1.1 Why loss of precision is good

There are two general approaches to general data compression:

- **Lossless**: The reconstructed data (after compression and subsequent decompression) are bit-for-bit identical to the original data.
- **Lossy**: The reconstructed data are not the same as the original.

In scientific data asking for lossless compression often is counter-productive. For example a meteorological model may produce temperature values as 4-byte floats or 8-byte floats, which store about 7 or 14 decimal digits respectively. However the input to the model may be temperature values from sensors that are valid to only 2 or 3 digits.

The remaining non-significant digits of the model often are statistically similar to random noise. Since random noise cannot be compressed, attempts to compress such data generally result in poor compression ratios.

If we use a lossy compression system to discard the useless random bits, we can achieve far better compression with no loss to the significant part of the data.

3.1.2 Metrics

Let v_i, for $0 \leq i \leq n-1$, represent the n true values, and r_i represent the n values reconstructed after compression and decompression.

Some common metrics for data and image compression error are:

1. Max absolute error: $E = max_i |r_i - v_i|$
2. MSE: mean squared error $S = 1/n \sum (r_i - v_i)^2$
3. PSNR: peak signal to noise ratio $= 10 \cdot log_{10}(M^2/S)$ where M is the max possible value. PSNR is a measure of accuracy as opposed to error. Note that a large word size (large M) makes the PSNR value artificially high.
4. Various visual perception measures.

Most image compression studies use metrics 2 - 4. However metrics 2 - 4 can trade small errors over many values for huge errors over a few values. There is no guarantee that ***all*** values are reasonably accurate, as a few values may have enormous errors.

Metric 1, the max absolute error, is the only metric in this list designed to make sure ***all*** values are within a specified accuracy bounds.

The reason the choice of error metric is important is that it strongly affects the architecture choices that follow.

If we use metric 1, the max absolute error, our choices for a software solution become much narrower – we must use lossless integer wavelets. If we were to use a metric like MSE, we could use lossy floating point wavelets and gain better compression, at the expense of accuracy.

Relative error measures won't work, because the question becomes "relative to what". Typically it's relative to the maximum absolute data value, but if a few noisy values have huge values then the error tolerance becomes unreasonably large.

3.1.3 Compression ratio

In all the files tested in this study, the original metadata claimed that the original values, before compression, had a specific format – typically 4 byte floating point values. When calculating compression ratios, we used that declared length as a reference. For example, if the data were declared as 4 byte floats,

$$\text{inputLength} = 4 \times \text{numberOfValues} \quad \text{(in bytes)}$$

$$\text{outputLength} = \text{outputFileLength} \quad \text{(in bytes)}$$

3.1.4 Compression ratio - old definition

Many comparisons of compression software define the compression ratio as

$$r_{\text{old}} = \text{InputLength}/\text{OutputLength}$$

So a ratio of 3 would indicate the compressed file takes 1/3 the space of the original, and in general larger values of r_{old} are better.

However, this definition of compression ratio is misleading. Very large values of r_{old}, such as 100 or 1000, indicate great compression but generally they occur on files that are of little interest. These values usually occur on small to mid size files that contain all zeros or missing values.

Since our interest is estimating bandwidth and storage requirements, such files are unimportant for our purposes. These files compress to a tiny size and have minuscule effect on the network and storage load.

Here the primary interest is in large files that compress poorly.

A second issue with using the r_{old} definition is that the large values for unimportant files can result in misleading statistical analysis results. Since most statistical procedures involve minimization of a mean squared error, or occasionally of a mean absolute error, the minimization will be strongly influenced by the large r_{old} values.

A final issue with using the r_{old} definition is that when plotting r_{old}, all the significant files (with small r_{old}) get lost along the base line of graph, while the unimportant files filled with missing values get prominent placement.

3.1.5 Compression ratio - new definition

We define the compression ratio to be

$$r = \text{OutputLength}/\text{InputLength}$$

So lower values of r are better, and a ratio of 1/3 represents a file that compressed to 1/3 it's original size.

Using this definition the important files – those having poor compression – show up as having large r values.

3.2 Compression subsystem criteria

The following are our criteria for a compression system for meteorological, scientific, and engineering data. When used on a wide variety of meteorological datasets, the system should:

- Maintain a specified max absolute error metric, as discussed in section 3.1.2
- Provide good compression
- Run reasonably fast
- Have no IP (intellectual property) issues like patents or restrictive licenses.
- Support self contained chunks. A chunk is a contiguous hyperrectangle subset of a gridded dataset. Typically a chunk has roughly 10^4 to 10^7 elements of data.
- Work well on a modern workstation or server with large main memory. We are not planning to support embedded processors with constrained memory.
- Support resolution of at least 1 part in 2^{31}
- Handle a variety of user-defined missing values, for example NaN, +/-infinity, -9999, etc.
- Allow implementations in Java and C
- Keep both the API and the software implementation as simple as possible commensurate with the above goals. This implies:
  - No extraneous features like region of interest compression
  - No user selectable features or user tunable values, aside from the specified accuracy criterion.

4 Compression pipeline

The overall compression process consists of three overall stages. Each stage contains one or more transforms. The overall stages are:

1. **Reduce variance and randomness.** This stage does not change the number of data values; it simply transforms them to reduce the variance and randomness. It includes the following transforms:
   - Lossless scanning transform
   - Lossless predictive transform
   - Lossy quantization
   - Lossless integer discrete wavelet transform
2. **Reduce the number of data values.** This stage may include:
   - Lossless Lempel Ziv dictionary transform
   - Lossless run length encoding

3. **Reduce the number of bits used to represent each value.** This stage is often called entropy encoding. There are a wide variety of choices for entropy encoding. We have chosen:
    - Lossless Huffman encoding

The overall compression process is a pipeline consisting of the above transforms. The decompression process is just the reverse.

A detailed description of each transform follows.

4.1 Lossless scanning transform

The scanning transform converts multidimensional data to one dimensional data. The transform from multi-dimensional to 1-dimensional data could happen at many places in the pipeline. We chose to put it first to simplify the software development. All subsequent stages of the pipeline only need to deal with 1-dimensional data.

There are many possible approaches to the scanning transform:

- Raster
- Boustrophedonic (winding back and forth)
- Diagonal
- Space filling curves - many possibilities

Scanning also may involve statistical measures of dimensional variance. For example, traversing dimensions having low variance more rapidly than those having high variance generally results in better compression.

We are investigating several of these methods and will choose only one for the final implementation.

4.2 Lossless predictive transform

There are many approaches to predictive transforms. If we were dealing with the original n-dimensional data we would use an n-dimensional predictive transform, described in 4.2.1.

We chose to use the scanner, converting from n to 1 dimensional data, as the first step of the pipeline. So the predictive transform need only deal with 1 dimensional data, as described in 4.2.2.

4.2.1 Lossless n-dimensional predictive transform

This is similar to the standard practice of replacing data with deltas from the previous value. But in this case we take into account multidimensional data and data patterns. For example in 3 dimensions, let $v_{i,j,k}$ be the value at point (i, j, k). We can develop a predictor $P_{i,j,k}$ to predict $v_{i,j,k}$ based on the nearby values $v_{i-1,j,k}, v_{i,j-1,k}, v_{i,j,k-1}, v_{i-1,j-1,k}$, etc. Now we can replace the values

$v_{i,j,k}$ with the differences $d_{i,j,k} = P_{i,j,k} - v_{i,j,k}$. In areas where the predictor is accurate the values $d_{i,j,k}$ will be close to 0, allowing better compression.

One minor difficulty is that the predictor P is only defined on the interior of the volume. The edges and faces of the volume must be handled with separate predictors using fewer dimensions. To decode an n dimensional volume, we would ...

- Using a 1 dimensional predictor for each edge, decode the edges (1 dimensional objects)
- Using the edge values and a 2 dimensional predictor for each face, decode the faces (2 dimensional objects).
- Using the edge and face values and a 3 dimensional predictor for the volume interior, decode the volume interior (a 3 dimensional object).
- For objects with over 3 dimensions, we would continue up the chain, decoding k dimensional objects using k dimensional predictors and the previously decoded values on the $k-1$ dimensional faces.

4.2.2 Lossless 1-dimensional predictive transform

The 1-dimensional predictive transform is similar to replacing the data values with the differences from the previous value, except in this case we use a least squares regression to create the prediction value.

We create a prediction P_i for each value v_i based on the previous values $v_{i-1}, v_{i-2}, \ldots$ and replace each value by the delta

$$d_i = v_i - P_i$$

Initial tests show that a simple one dimensional linear predictor is nearly as good as the least squares optimization model.

4.3 Lossy quantization

Quantization is the conversion of floating point values to integer values. Because of our criterion for tight control over the max absolute error, quantization is the *only* part of the entire pipeline that introduces loss of accuracy.

4.4 Lossless integer discrete wavelet transform

There are many approaches to integer wavelet transforms. The integer wavelet field started in 1996 with Calderbank, Daubechies, and Sweldens [10]. Adams provided surveys of the integer wavelet techniques contributing to JPEG2000 in [2][1].

Two commonly used enhancements to wavelet based image compression are Embedded Zerotrees (EZW) by Shapiro [37] and Set Partitioning in Hierarchical Trees (SPIHT) by Said and Pearlman [34]. Both EZW and SPIHT have been patented, so are not available for use within this project.

There have been numerous approaches to lossless and integer wavelet compression, such as [38], [4], [6], [33], [17], [59], [44], [60], and [15].

The recent papers of Tilo Strutz, [42], [41], [40], [43], generalize many approaches to the discrete wavelet transform, both for real and integer values. In addition, the approaches used by Strutz offer elegant solutions to two long standing problems in the discrete wavelet transform.

1. **Lengths must be a power of 2.**
   Most wavelet transform algorithms are designed to work only on lengths that are an integer power of 2 – for example, images that are 512 or 1024 pixels on each side. Typically people extend such algorithms to process other sizes by either:
   - Extending the image boundaries to the next larger power of 2. This can impact both CPU time and memory performance, as a 520 x 520 image would be extended to 1024 x 1024, nearly quadrupling its size.
   - Subdividing the image into smaller blocks whose edge lengths are powers of 2. This involves extra software complexity, negatively impacts the compression ratio, and can introduce artifacts along the block boundaries.

   The approaches used by Strutz allow an elegant generalization beyond the standard power of 2 constraints.
2. **Wavelets create edge effects at boundaries.**
   In all but the simplest Haar wavelet there are issues dealing with the wavelet at the boundary of the region to be transformed. There are a variety of methods to deal with these issues, ranging from moderately complex to definitely complex. All the approaches have various trade offs of processing time and accuracy. The approach of Strutz for handling boundary conditions is both elegant and general.

4.5 Lossless Lempel Ziv dictionary transform

The LZ compression system converts sequences of values to a single value using an adaptive dictionary of sequences.

There are many variants of the LZ algorithm.

- LZ77
- LZW
- LZMA
- Statistical LZ
- Adaptive LZ
- and many more

Some of them are encumbered by patents or other intellectual property issues. We have chosen to use an extension of the LZW algorithm. The standard LZW algorithm assumes a fixed and known symbol set – typically the 256 possible values for an 8-bit byte. In our case the symbol set is all possible integers, 2^{32} or 2^{64} possible values. We have adapted the LZW algorithm to handle an essentially infinite set of possible values.

4.6 Lossless run length encoding

Run length encoding (RLE) changes a sequence of values by replacing subsequences of consecutive identical values with a control sequence indicating the repetition. RLE requires two types of control information:

- The following n symbols are identical and their value is x.
- The following n symbols are *not* identical and their values are xyz....

There are a handful of decisions to be made around the nature of the control information. The number of bits devoted to each control symbol affects the overall compression ratio.

4.7 Lossless entropy transform

The entropy transform replaces commonly occurring symbols with short bit strings and rarely occurring symbols with long bit strings. While there are many possible choices for the entropy transform, we have chosen to use the Huffman because it is relatively fast, effective, and straight forward to implement. The Huffman transform is provably optimal for non-correlated sequences of symbols.

5 Test data

We chose a variety of datasets and variables for testing the compression / decompression system. We chose:

- Datasets and variables of importance to the FAA
- Datasets large enough to cause concern. For example, we omitted METARs because they're small.
- Variables with a variety of precisions
- Variables with a variety of spatial characteristics.

The datasets and variables we used are:

5.1 Rapid Refresh (RAP) Model

Descriptive information:

URL	http://www.nco.ncep.noaa.gov/pmb/products/rap/
Description	Regional, CONUS, pressure levels, 13-km resolution
Grid	x: 451, y: 337, z: 37
Projection	Lambert Conformal

Variables:

Name	**Long name**	**Units**	**Resolution**
HGT	Geopotential Height	gpm	0.125
TMP	Temperature	K	0.125
RH	Relative Humidity	%	1.0
UGRD	U-Component of Wind	m/s	0.125
VGRD	V-Component of Wind	m/s	0.125
VVEL	Vertical Velocity (Pressure)	Pa/s	0.03125

5.2 Global Forecast System (GFS) Model

Descriptive information:

URL	http://www.nco.ncep.noaa.gov/pmb/products/gfs/
Description	Global longitude-latitude grid, 0.5 degree resolution
Grid	x: 720, y: 361, z: 21
Projection	Longitude latitude

Variables:

Name	**Long name**	**Units**	**Resolution**
HGT	Geopotential Height	gpm	1.e-3
TMP	Temperature	K	1.e-1
RH	Relative Humidity	%	1.0
SPFH	Specific Humidity	kg/kg	1.e-5
VVEL	Vertical Velocity (Pressure)	Pa/s	1.e-3
UGRD	U-Component of Wind	m/s	1.e-2
VGRD	V-Component of Wind	m/s	1.e-2
ABSV	Absolute Vorticity	1/s	1.e-6
CLWMR	Cloud Mixing Ratio	kg/kg	1.e-7

5.3 NEXRAD (Next Generation Weather Radar)

Descriptive information:

URL	http://www.roc.noaa.gov/WSR88D/
Description	Doppler radar within the CONUS
Grid	azimuth: 360, gate: 230
Projection	Radar polar

Variables:

Name	**Long name**	**Units**	**Resolution**
BREF	Base Reflectivity	dbZ	5.0

6 Test results

6.1 Base case: gzip

As a base case, we tested ordinary gzip compression (lossless) on each variable. The results are shown in table 1. Here $\boldsymbol{r_{gzip}}$ is the ratio gzippedLength / originalLength, so smaller values of $\boldsymbol{r_{gzip}}$ indicate better compression.

Table 1: Compression with lossless gzip

DSN	**Var**	$\boldsymbol{r_{gzip}}$
RAP	HGT	0.953
RAP	TMP	0.933
RAP	RH	0.936
RAP	UGRD	0.964
RAP	VGRD	0.966
RAP	VVEL	0.970
GFS	HGT	0.994
GFS	TMP	0.990
GFS	RH	0.975
GFS	SPFH	0.998
GFS	VVEL	0.998
GFS	UGRD	0.998
GFS	VGRD	0.998
GFS	ABSV	0.998
GFS	CLWMR	0.993
NEXRAD	BREF1	0.607

Clearly for these datasets, lossless gzip compression isn't effective. This provides a good example of the need for controlled loss of precision, as described in section 3.1.1.

6.2 Variations on the compression pipeline

When testing the prototype compression pipeline, each file was tested as follows:

- If a file contains multiple variables, extract just the variable of interest to a new single-variable file of the same type. For example, when using a RAP Grib2 file containing many variables, this meant creating a new Grib2 file with just the TMP variable. The length of the single-variable file is used as the "original file length" in the calculation of $\boldsymbol{r_{orig}}$, below.
- Convert the single-variable file to uncompressed NetCDF format to provide a common format for compression. This also combines the separate 2-dimensional Grib2 records into a single 3 or 4 dimensional volume.
- Determine the max absolute error. By examining the details of the Grib2 encoding determine the Grib2 resolution. Our max absolute error is half that value. For example the RAP TMP field is encoded in Grib2 at a resolution of 1/8 Kelvin, so the max absolute error in our quantization step would be 1/16 Kelvin.
- Compress the NetCDF file. We tested each file with eight variations of the compression pipeline:
  - With and without wavelet compression
  - With and without run length encoding
  - With and without Lempel-Ziv compression
- Measure the length of the compressed file. This is used as the "test compressed file length" in the calculation of $\boldsymbol{r_{new}}$, below.

Table 2 shows the test results for 8 variations on the compression pipeline, for each variable.

The columns are:

DSN	Dataset name
Var	Variable name
Wv	Wavelet compression, n/y
RLE	Run length encoding compression, n/y
LZ	Lempel-Ziv compression, n/y
$\boldsymbol{r_{orig}}$	Original file compression factor. This is the length of the original file (typically in Grib2 format) divided by the calculated uncompressed length of the file, $4 * totalNumElements$.
$\boldsymbol{r_{new}}$	New file compression factor. This is the length of the compressed test file divided by the calculated uncompressed length of the file, $4 * totalNumElements$.
$\boldsymbol{r_{new}/r_{orig}}$	The ratio of the test compressed file length to the original file length. If this value is less than one, the new compression is more efficient than the existing compression method.
$\boldsymbol{t_{enc}}$	Encoding (compression) time in seconds
$\boldsymbol{t_{dec}}$	Decoding (decompression) time in seconds

The rows with the minimum value of $\boldsymbol{r_{new}}$ for each variable are shown in **bold**, and the ratio $\boldsymbol{r_{new}/r_{orig}}$ is highlighted.

Table 2: Compression / decompression results

DSN	**Var**	**Wv**	**RLE**	**LZ**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
RAP	HGT	n	n	n	0.17	0.59	3.41	0.23	0.18
RAP	HGT	n	n	y	0.17	1.11	6.38	1.39	0.45
RAP	HGT	n	y	n	0.17	0.59	3.39	0.26	0.21
RAP	HGT	n	y	y	0.17	1.18	6.78	1.82	0.63
RAP	**HGT**	**y**	**n**	**n**	**0.17**	**0.39**	**2.23**	**0.27**	**0.17**
RAP	HGT	y	n	y	0.17	1.08	6.22	2.37	0.60
RAP	HGT	y	y	n	0.17	0.41	2.35	0.30	0.23
RAP	HGT	y	y	y	0.17	1.12	6.45	2.33	0.69
RAP	TMP	n	n	n	0.19	0.40	2.10	0.21	0.12
RAP	TMP	n	n	y	0.19	1.13	5.96	1.13	0.36
RAP	TMP	n	y	n	0.19	0.44	2.32	0.25	0.15
RAP	TMP	n	y	y	0.19	1.21	6.36	1.15	0.41
RAP	**TMP**	**y**	**n**	**n**	**0.19**	**0.40**	**2.09**	**0.25**	**0.15**
RAP	TMP	y	n	y	0.19	1.29	6.76	1.47	0.46
RAP	TMP	y	y	n	0.19	0.45	2.36	0.29	0.20
RAP	TMP	y	y	y	0.19	1.36	7.14	1.62	0.57
RAP	RH	n	n	n	0.22	0.44	2.00	0.21	0.14
RAP	RH	n	n	y	0.22	1.27	5.74	1.37	0.48
RAP	RH	n	y	n	0.22	0.47	2.11	0.25	0.16
RAP	RH	n	y	y	0.22	1.32	5.96	1.64	0.55
RAP	**RH**	**y**	**n**	**n**	**0.22**	**0.43**	**1.96**	**0.26**	**0.17**
RAP	RH	y	n	y	0.22	1.30	5.90	1.78	0.54
RAP	RH	y	y	n	0.22	0.46	2.08	0.30	0.20
RAP	RH	y	y	y	0.22	1.35	6.10	2.07	0.69
RAP	UGRD	n	n	n	0.15	0.36	2.47	0.21	0.15
RAP	UGRD	n	n	y	0.15	0.94	6.35	1.31	0.46
RAP	UGRD	n	y	n	0.15	0.38	2.55	0.25	0.18
RAP	UGRD	n	y	y	0.15	0.98	6.65	1.56	0.64
RAP	**UGRD**	**y**	**n**	**n**	**0.15**	**0.30**	**2.05**	**0.26**	**0.17**
RAP	UGRD	y	n	y	0.15	0.92	6.22	1.79	0.58
RAP	UGRD	y	y	n	0.15	0.33	2.21	0.30	0.21
RAP	UGRD	y	y	y	0.15	0.95	6.46	1.96	0.66
RAP	VGRD	n	n	n	0.14	0.33	2.36	0.20	0.14
RAP	VGRD	n	n	y	0.14	0.88	6.38	1.35	0.47
RAP	VGRD	n	y	n	0.14	0.34	2.46	0.25	0.17
RAP	VGRD	n	y	y	0.14	0.92	6.63	1.67	0.64
RAP	**VGRD**	**y**	**n**	**n**	**0.14**	**0.28**	**2.04**	**0.25**	**0.17**
RAP	VGRD	y	n	y	0.14	0.86	6.23	1.73	0.54
RAP	VGRD	y	y	n	0.14	0.30	2.19	0.30	0.20
RAP	VGRD	y	y	y	0.14	0.89	6.45	1.92	0.64

... continued on next page

... continued from previous page

DSN	Var	Wv	RLE	LZ	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
RAP	**VVEL**	**n**	**n**	**n**	**0.14**	**0.26**	**1.84**	**0.21**	**0.13**
RAP	VVEL	n	n	y	0.14	0.77	5.45	1.46	0.49
RAP	VVEL	n	y	n	0.14	0.28	1.95	0.25	0.17
RAP	VVEL	n	y	y	0.14	0.80	5.64	1.59	0.58
RAP	VVEL	y	n	n	0.14	0.26	1.86	0.25	0.16
RAP	VVEL	y	n	y	0.14	0.78	5.52	1.77	0.54
RAP	VVEL	y	y	n	0.14	0.28	1.97	0.29	0.20
RAP	VVEL	y	y	y	0.14	0.81	5.72	1.99	0.61
GFS	HGT	n	n	n	0.58	1.17	2.00	0.44	0.42
GFS	HGT	n	n	y	0.58	1.64	2.82	34.78	21.74
GFS	HGT	n	y	n	0.58	1.16	2.00	0.45	0.45
GFS	HGT	n	y	y	0.58	1.64	2.82	38.28	23.66
GFS	HGT	y	n	n	0.58	0.98	1.68	0.41	0.40
GFS	HGT	y	n	y	0.58	1.74	2.98	40.98	21.55
GFS	**HGT**	**y**	**y**	**n**	**0.58**	**0.98**	**1.68**	**0.42**	**0.43**
GFS	HGT	y	y	y	0.58	1.74	2.98	42.28	24.36
GFS	**TMP**	**n**	**n**	**n**	**0.30**	**0.48**	**1.61**	**0.22**	**0.14**
GFS	TMP	n	n	y	0.30	1.37	4.59	1.72	0.54
GFS	TMP	n	y	n	0.30	0.49	1.66	0.25	0.17
GFS	TMP	n	y	y	0.30	1.40	4.70	1.95	0.63
GFS	TMP	y	n	n	0.30	0.48	1.62	0.27	0.17
GFS	TMP	y	n	y	0.30	1.39	4.67	2.04	0.58
GFS	TMP	y	y	n	0.30	0.50	1.68	0.30	0.21
GFS	TMP	y	y	y	0.30	1.43	4.79	2.15	0.70
GFS	RH	n	n	n	0.56	1.02	1.83	0.20	0.14
GFS	RH	n	n	y	0.56	2.16	3.89	1.64	0.50
GFS	**RH**	**n**	**y**	**n**	**0.56**	**0.83**	**1.50**	**0.22**	**0.15**
GFS	RH	n	y	y	0.56	2.20	3.96	1.62	0.58
GFS	RH	y	n	n	0.56	0.94	1.69	0.25	0.17
GFS	RH	y	n	y	0.56	2.21	3.99	2.33	0.59
GFS	RH	y	y	n	0.56	0.84	1.51	0.28	0.19
GFS	RH	y	y	y	0.56	2.24	4.04	2.45	0.72
GFS	SPFH	n	n	n	0.59	0.32	0.54	0.20	0.16
GFS	SPFH	n	n	y	0.59	0.59	1.00	2.05	0.64
GFS	SPFH	n	y	n	0.59	0.26	0.44	0.23	0.17
GFS	SPFH	n	y	y	0.59	0.60	1.01	4.28	1.03
GFS	SPFH	y	n	n	0.59	0.29	0.49	0.25	0.18
GFS	SPFH	y	n	y	0.59	0.60	1.01	2.15	0.80
GFS	**SPFH**	**y**	**y**	**n**	**0.59**	**0.25**	**0.43**	**0.27**	**0.21**
GFS	SPFH	y	y	y	0.59	0.61	1.03	2.84	0.91

... continued on next page

... continued from previous page

DSN	**Var**	**Wv**	**RLE**	**LZ**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
GFS	VVEL	n	n	n	0.33	0.44	1.34	0.22	0.22
GFS	VVEL	n	n	y	0.33	0.82	2.50	6.73	3.48
GFS	VVEL	n	y	n	0.33	0.39	1.20	0.25	0.23
GFS	VVEL	n	y	y	0.33	0.82	2.50	7.71	4.10
GFS	VVEL	y	n	n	0.33	0.40	1.23	0.28	0.24
GFS	VVEL	y	n	y	0.33	0.77	2.35	6.91	3.71
GFS	**VVEL**	**y**	**y**	**n**	**0.33**	**0.37**	**1.12**	**0.29**	**0.27**
GFS	VVEL	y	y	y	0.33	0.77	2.35	7.74	3.97
GFS	UGRD	n	n	n	0.31	0.51	1.67	0.23	0.23
GFS	UGRD	n	n	y	0.31	1.12	3.66	8.56	4.07
GFS	UGRD	n	y	n	0.31	0.51	1.67	0.27	0.25
GFS	UGRD	n	y	y	0.31	1.12	3.66	8.54	3.27
GFS	**UGRD**	**y**	**n**	**n**	**0.31**	**0.49**	**1.60**	**0.27**	**0.25**
GFS	UGRD	y	n	y	0.31	1.09	3.56	7.78	2.93
GFS	UGRD	y	y	n	0.31	0.49	1.60	0.33	0.31
GFS	UGRD	y	y	y	0.31	1.09	3.56	8.77	3.31
GFS	VGRD	n	n	n	0.28	0.49	1.71	0.23	0.23
GFS	VGRD	n	n	y	0.28	1.06	3.73	11.43	3.72
GFS	VGRD	n	y	n	0.28	0.49	1.72	0.27	0.27
GFS	VGRD	n	y	y	0.28	1.06	3.73	11.16	4.75
GFS	**VGRD**	**y**	**n**	**n**	**0.28**	**0.46**	**1.61**	**0.28**	**0.25**
GFS	VGRD	y	n	y	0.28	1.03	3.62	8.33	3.78
GFS	VGRD	y	y	n	0.28	0.46	1.62	0.32	0.29
GFS	VGRD	y	y	y	0.28	1.03	3.63	9.23	3.54
GFS	ABSV	n	n	n	0.23	0.35	1.48	0.22	0.18
GFS	ABSV	n	n	y	0.23	0.80	3.39	3.03	0.88
GFS	ABSV	n	y	n	0.23	0.35	1.48	0.26	0.22
GFS	ABSV	n	y	y	0.23	0.80	3.41	4.26	1.26
GFS	**ABSV**	**y**	**n**	**n**	**0.23**	**0.32**	**1.36**	**0.27**	**0.21**
GFS	ABSV	y	n	y	0.23	0.77	3.26	3.09	0.97
GFS	ABSV	y	y	n	0.23	0.32	1.37	0.31	0.25
GFS	ABSV	y	y	y	0.23	0.77	3.29	4.41	1.34
GFS	CLWMR	n	n	n	0.31	0.84	2.66	0.24	0.15
GFS	CLWMR	n	n	y	0.31	1.05	3.34	1.18	0.71
GFS	**CLWMR**	**n**	**y**	**n**	**0.31**	**0.58**	**1.84**	**0.23**	**0.14**
GFS	CLWMR	n	y	y	0.31	1.03	3.28	1.65	0.53
GFS	CLWMR	y	n	n	0.31	0.88	2.81	0.29	0.19
GFS	CLWMR	y	n	y	0.31	1.38	4.40	2.90	1.12
GFS	CLWMR	y	y	n	0.31	0.66	2.10	0.28	0.19
GFS	CLWMR	y	y	y	0.31	1.35	4.28	3.35	1.17

... continued on next page

... continued from previous page

DSN	**Var**	**Wv**	**RLE**	**LZ**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
NEXRAD	BREF1	n	n	n	1.00	1.08	1.08	2.05	0.54
NEXRAD	BREF1	n	n	y	1.00	3.30	3.30	13.07	5.89
NEXRAD	**BREF1**	**n**	**y**	**n**	**1.00**	**0.98**	**0.98**	**2.16**	**0.59**
NEXRAD	BREF1	n	y	y	1.00	3.49	3.49	13.01	6.24
NEXRAD	BREF1	y	n	n	1.00	1.33	1.33	2.51	0.68
NEXRAD	BREF1	y	n	y	1.00	4.99	4.99	14.72	7.13
NEXRAD	BREF1	y	y	n	1.00	1.39	1.39	2.63	0.71
NEXRAD	BREF1	y	y	y	1.00	5.00	5.00	13.02	7.46

For each variable in the table above, the configuration Wv=y, RLE=n, LZ=n either has the minimum, or very close to the minimum, value of r_{new}, which implies the minimum of r_{new}/r_{orig}.

This implies that the RLE and LZ aren't effective in improving compression. The wavelet transform has already reduced the redundancy so that there is little further work that can be done by LZ and RLE. Or another way of looking at it, the output of the wavelet transform is sufficiently random that the RLE and LZ can find few repeating values or sequences to compress.

6.3 Tests with different wavelets

Table 3 shows the test results for 26 wavelet types, for each variable. The column “**Wnum**” is the wavelet number from Strutz [42].

The rows with the minimum value of r_{new} for each variable are shown in **bold**, and the ratio r_{new}/r_{orig} is highlighted.

Table 3: Compression / decompression results

DSN	Var	Wnum	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
RAP	**HGT**	**0**	**0.17**	**0.39**	**2.23**	**0.26**	**0.17**
RAP	HGT	1	0.17	0.42	2.44	0.31	0.20
RAP	HGT	2	0.17	0.42	2.42	0.27	0.19
RAP	HGT	3	0.17	0.42	2.40	0.27	0.19
RAP	HGT	4	0.17	0.42	2.41	0.28	0.19
RAP	HGT	5	0.17	0.42	2.40	0.28	0.20
RAP	HGT	6	0.17	0.42	2.41	0.30	0.20
RAP	HGT	7	0.17	0.42	2.42	0.29	0.21
RAP	HGT	8	0.17	0.41	2.37	0.28	0.19
RAP	HGT	9	0.17	0.42	2.43	0.29	0.20
RAP	HGT	10	0.17	0.43	2.46	0.28	0.21
RAP	HGT	11	0.17	0.41	2.34	0.28	0.22
RAP	HGT	12	0.17	0.41	2.36	0.28	0.19
RAP	HGT	13	0.17	0.42	2.38	0.28	0.19
RAP	HGT	14	0.17	0.41	2.37	0.28	0.19
RAP	HGT	15	0.17	0.42	2.41	0.29	0.20
RAP	HGT	16	0.17	0.42	2.42	0.28	0.20
RAP	HGT	17	0.17	0.43	2.48	0.28	0.20
RAP	HGT	18	0.17	0.45	2.56	0.29	0.20
RAP	HGT	19	0.17	0.41	2.37	0.29	0.20
RAP	HGT	20	0.17	0.41	2.34	0.28	0.20
RAP	HGT	21	0.17	0.41	2.38	0.29	0.20
RAP	HGT	22	0.17	0.41	2.38	0.33	0.20
RAP	HGT	23	0.17	0.41	2.38	0.28	0.19
RAP	HGT	24	0.17	0.41	2.38	0.28	0.20
RAP	HGT	25	0.17	0.41	2.38	0.28	0.20
RAP	**TMP**	**0**	**0.19**	**0.40**	**2.09**	**0.27**	**0.16**
RAP	TMP	1	0.19	0.43	2.26	0.27	0.17
RAP	TMP	2	0.19	0.42	2.23	0.28	0.17
RAP	TMP	3	0.19	0.43	2.26	0.32	0.17
RAP	TMP	4	0.19	0.42	2.23	0.27	0.17
RAP	TMP	5	0.19	0.43	2.26	0.28	0.17

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
RAP	TMP	6	0.19	0.42	2.22	0.27	0.18
RAP	TMP	7	0.19	0.43	2.24	0.27	0.17
RAP	TMP	8	0.19	0.41	2.15	0.27	0.18
RAP	TMP	9	0.19	0.43	2.28	0.27	0.17
RAP	TMP	10	0.19	0.44	2.33	0.27	0.18
RAP	TMP	11	0.19	0.42	2.22	0.27	0.17
RAP	TMP	12	0.19	0.42	2.21	0.27	0.17
RAP	TMP	13	0.19	0.42	2.23	0.30	0.17
RAP	TMP	14	0.19	0.42	2.20	0.27	0.17
RAP	TMP	15	0.19	0.43	2.28	0.27	0.18
RAP	TMP	16	0.19	0.43	2.24	0.28	0.17
RAP	TMP	17	0.19	0.45	2.39	0.27	0.17
RAP	TMP	18	0.19	0.44	2.31	0.27	0.17
RAP	TMP	19	0.19	0.42	2.22	0.27	0.17
RAP	TMP	20	0.19	0.41	2.17	0.27	0.17
RAP	TMP	21	0.19	0.42	2.23	0.26	0.17
RAP	TMP	22	0.19	0.42	2.22	0.26	0.17
RAP	TMP	23	0.19	0.42	2.22	0.27	0.17
RAP	TMP	24	0.19	0.42	2.23	0.26	0.17
RAP	TMP	25	0.19	0.42	2.23	0.27	0.17
RAP	**RH**	**0**	**0.22**	**0.43**	**1.96**	**0.26**	**0.17**
RAP	RH	1	0.22	0.45	2.03	0.27	0.18
RAP	RH	2	0.22	0.45	2.02	0.27	0.19
RAP	RH	3	0.22	0.45	2.02	0.27	0.18
RAP	RH	4	0.22	0.44	2.01	0.29	0.18
RAP	RH	5	0.22	0.45	2.02	0.27	0.19
RAP	RH	6	0.22	0.44	2.02	0.27	0.18
RAP	RH	7	0.22	0.45	2.02	0.30	0.18
RAP	RH	8	0.22	0.44	1.98	0.27	0.18
RAP	RH	9	0.22	0.45	2.04	0.27	0.19
RAP	RH	10	0.22	0.46	2.06	0.27	0.18
RAP	RH	11	0.22	0.45	2.03	0.27	0.18
RAP	RH	12	0.22	0.44	2.01	0.27	0.19
RAP	RH	13	0.22	0.44	2.00	0.27	0.19
RAP	RH	14	0.22	0.44	2.00	0.27	0.18
RAP	RH	15	0.22	0.45	2.03	0.27	0.18
RAP	RH	16	0.22	0.45	2.02	0.27	0.18
RAP	RH	17	0.22	0.46	2.09	0.28	0.18
RAP	RH	18	0.22	0.46	2.06	0.27	0.19
RAP	RH	19	0.22	0.44	2.00	0.27	0.18
RAP	RH	20	0.22	0.44	1.98	0.28	0.18

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
RAP	RH	21	0.22	0.44	2.01	0.27	0.18
RAP	RH	22	0.22	0.44	2.00	0.27	0.18
RAP	RH	23	0.22	0.44	2.00	0.27	0.18
RAP	RH	24	0.22	0.44	2.01	0.28	0.18
RAP	RH	25	0.22	0.44	2.01	0.27	0.18
RAP	**UGRD**	**0**	**0.15**	**0.30**	**2.05**	**0.26**	**0.17**
RAP	UGRD	1	0.15	0.32	2.18	0.30	0.18
RAP	UGRD	2	0.15	0.32	2.17	0.27	0.19
RAP	UGRD	3	0.15	0.32	2.15	0.27	0.18
RAP	UGRD	4	0.15	0.32	2.15	0.27	0.18
RAP	UGRD	5	0.15	0.32	2.15	0.27	0.18
RAP	UGRD	6	0.15	0.32	2.15	0.27	0.20
RAP	UGRD	7	0.15	0.32	2.15	0.27	0.18
RAP	UGRD	8	0.15	0.31	2.12	0.28	0.18
RAP	UGRD	9	0.15	0.32	2.17	0.27	0.18
RAP	UGRD	10	0.15	0.32	2.19	0.26	0.21
RAP	UGRD	11	0.15	0.32	2.17	0.27	0.19
RAP	UGRD	12	0.15	0.31	2.13	0.28	0.18
RAP	UGRD	13	0.15	0.31	2.13	0.27	0.18
RAP	UGRD	14	0.15	0.31	2.13	0.29	0.18
RAP	UGRD	15	0.15	0.32	2.16	0.27	0.19
RAP	UGRD	16	0.15	0.32	2.15	0.28	0.19
RAP	UGRD	17	0.15	0.33	2.22	0.28	0.19
RAP	UGRD	18	0.15	0.33	2.22	0.26	0.18
RAP	UGRD	19	0.15	0.31	2.12	0.27	0.18
RAP	UGRD	20	0.15	0.31	2.13	0.26	0.18
RAP	UGRD	21	0.15	0.31	2.12	0.28	0.18
RAP	UGRD	22	0.15	0.31	2.12	0.28	0.19
RAP	UGRD	23	0.15	0.31	2.12	0.27	0.18
RAP	UGRD	24	0.15	0.31	2.12	0.27	0.18
RAP	UGRD	25	0.15	0.31	2.12	0.27	0.18
RAP	**VGRD**	**0**	**0.14**	**0.28**	**2.04**	**0.25**	**0.17**
RAP	VGRD	1	0.14	0.30	2.15	0.26	0.18
RAP	VGRD	2	0.14	0.30	2.14	0.26	0.17
RAP	VGRD	3	0.14	0.29	2.13	0.26	0.17
RAP	VGRD	4	0.14	0.29	2.13	0.30	0.18
RAP	VGRD	5	0.14	0.30	2.14	0.26	0.18
RAP	VGRD	6	0.14	0.29	2.13	0.26	0.18
RAP	VGRD	7	0.14	0.30	2.14	0.26	0.18
RAP	VGRD	8	0.14	0.29	2.11	0.26	0.18
RAP	VGRD	9	0.14	0.30	2.16	0.26	0.18

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
RAP	VGRD	10	0.14	0.30	2.18	0.26	0.18
RAP	VGRD	11	0.14	0.30	2.15	0.26	0.19
RAP	VGRD	12	0.14	0.29	2.11	0.26	0.17
RAP	VGRD	13	0.14	0.29	2.11	0.26	0.18
RAP	VGRD	14	0.14	0.29	2.11	0.26	0.18
RAP	VGRD	15	0.14	0.30	2.15	0.27	0.18
RAP	VGRD	16	0.14	0.30	2.14	0.27	0.18
RAP	VGRD	17	0.14	0.31	2.21	0.27	0.18
RAP	VGRD	18	0.14	0.31	2.21	0.27	0.19
RAP	VGRD	19	0.14	0.29	2.10	0.27	0.18
RAP	VGRD	20	0.14	0.29	2.10	0.27	0.18
RAP	VGRD	21	0.14	0.29	2.11	0.26	0.18
RAP	VGRD	22	0.14	0.29	2.11	0.26	0.18
RAP	VGRD	23	0.14	0.29	2.11	0.27	0.18
RAP	VGRD	24	0.14	0.29	2.11	0.26	0.18
RAP	VGRD	25	0.14	0.29	2.11	0.27	0.18
RAP	**VVEL**	**0**	**0.14**	**0.26**	**1.86**	**0.26**	**0.17**
RAP	VVEL	1	0.14	0.27	1.89	0.27	0.18
RAP	VVEL	2	0.14	0.27	1.88	0.27	0.18
RAP	VVEL	3	0.14	0.27	1.90	0.27	0.18
RAP	VVEL	4	0.14	0.27	1.89	0.27	0.19
RAP	VVEL	5	0.14	0.27	1.91	0.27	0.18
RAP	VVEL	6	0.14	0.27	1.90	0.27	0.18
RAP	VVEL	7	0.14	0.27	1.91	0.27	0.19
RAP	VVEL	8	0.14	0.26	1.86	0.27	0.17
RAP	VVEL	9	0.14	0.27	1.93	0.27	0.18
RAP	VVEL	10	0.14	0.28	1.95	0.27	0.20
RAP	VVEL	11	0.14	0.27	1.91	0.28	0.18
RAP	VVEL	12	0.14	0.27	1.88	0.27	0.18
RAP	VVEL	13	0.14	0.27	1.89	0.27	0.18
RAP	VVEL	14	0.14	0.27	1.88	0.27	0.19
RAP	VVEL	15	0.14	0.27	1.92	0.27	0.18
RAP	VVEL	16	0.14	0.27	1.91	0.27	0.18
RAP	VVEL	17	0.14	0.28	1.97	0.26	0.18
RAP	VVEL	18	0.14	0.28	1.95	0.26	0.18
RAP	VVEL	19	0.14	0.27	1.89	0.27	0.18
RAP	VVEL	20	0.14	0.26	1.86	0.27	0.18
RAP	VVEL	21	0.14	0.27	1.90	0.27	0.18
RAP	VVEL	22	0.14	0.27	1.89	0.27	0.18
RAP	VVEL	23	0.14	0.27	1.89	0.27	0.18
RAP	VVEL	24	0.14	0.27	1.90	0.28	0.18

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
RAP	VVEL	25	0.14	0.27	1.90	0.27	0.18
GFS	**HGT**	**0**	**0.58**	**0.98**	**1.68**	**0.43**	**0.45**
GFS	HGT	1	0.58	1.04	1.79	0.48	0.44
GFS	HGT	2	0.58	1.02	1.76	0.50	0.47
GFS	HGT	3	0.58	1.01	1.73	0.49	0.45
GFS	HGT	4	0.58	1.00	1.72	0.49	0.49
GFS	HGT	5	0.58	1.00	1.71	0.48	0.49
GFS	HGT	6	0.58	0.99	1.71	0.45	0.47
GFS	HGT	7	0.58	0.99	1.71	0.44	0.41
GFS	HGT	8	0.58	0.99	1.70	0.45	0.47
GFS	HGT	9	0.58	0.99	1.70	0.44	0.43
GFS	HGT	10	0.58	0.98	1.69	0.46	0.47
GFS	HGT	11	0.58	0.99	1.71	0.47	0.48
GFS	HGT	12	0.58	1.01	1.73	0.50	0.52
GFS	HGT	13	0.58	1.00	1.72	0.48	0.52
GFS	HGT	14	0.58	1.00	1.72	0.49	0.46
GFS	HGT	15	0.58	0.99	1.71	0.43	0.41
GFS	HGT	16	0.58	0.99	1.71	0.47	0.49
GFS	HGT	17	0.58	0.99	1.70	0.46	0.48
GFS	HGT	18	0.58	0.98	1.69	0.45	0.47
GFS	HGT	19	0.58	1.00	1.72	0.47	0.50
GFS	HGT	20	0.58	1.00	1.72	0.47	0.47
GFS	HGT	21	0.58	1.00	1.72	0.48	0.47
GFS	HGT	22	0.58	1.00	1.72	0.47	0.48
GFS	HGT	23	0.58	1.00	1.72	0.49	0.50
GFS	HGT	24	0.58	1.00	1.72	0.48	0.51
GFS	HGT	25	0.58	1.00	1.72	0.49	0.50
GFS	**TMP**	**0**	**0.30**	**0.48**	**1.62**	**0.27**	**0.18**
GFS	TMP	1	0.30	0.49	1.64	0.28	0.19
GFS	TMP	2	0.30	0.49	1.63	0.29	0.19
GFS	TMP	3	0.30	0.49	1.64	0.29	0.19
GFS	TMP	4	0.30	0.49	1.65	0.30	0.20
GFS	TMP	5	0.30	0.49	1.65	0.28	0.19
GFS	TMP	6	0.30	0.49	1.65	0.28	0.19
GFS	TMP	7	0.30	0.49	1.66	0.28	0.19
GFS	TMP	8	0.30	0.48	1.63	0.28	0.19
GFS	TMP	9	0.30	0.50	1.67	0.29	0.19
GFS	TMP	10	0.30	0.50	1.69	0.28	0.19
GFS	TMP	11	0.30	0.49	1.65	0.27	0.19
GFS	TMP	12	0.30	0.49	1.64	0.29	0.20
GFS	TMP	13	0.30	0.49	1.64	0.29	0.20

... continued on next page

... continued from previous page

DSN	Var	Wnum	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
GFS	TMP	14	0.30	0.49	1.64	0.28	0.19
GFS	TMP	15	0.30	0.50	1.67	0.29	0.20
GFS	TMP	16	0.30	0.49	1.66	0.28	0.19
GFS	TMP	17	0.30	0.51	1.70	0.28	0.19
GFS	TMP	18	0.30	0.50	1.69	0.28	0.20
GFS	TMP	19	0.30	0.49	1.64	0.27	0.18
GFS	TMP	20	0.30	0.48	1.62	0.29	0.19
GFS	TMP	21	0.30	0.49	1.65	0.29	0.19
GFS	TMP	22	0.30	0.49	1.64	0.28	0.19
GFS	TMP	23	0.30	0.49	1.64	0.28	0.19
GFS	TMP	24	0.30	0.49	1.65	0.28	0.19
GFS	TMP	25	0.30	0.49	1.65	0.28	0.19
GFS	**RH**	**0**	**0.56**	**0.94**	**1.69**	**0.27**	**0.18**
GFS	RH	1	0.56	0.94	1.70	0.27	0.19
GFS	RH	2	0.56	0.95	1.72	0.27	0.20
GFS	RH	3	0.56	0.96	1.73	0.27	0.20
GFS	RH	4	0.56	0.96	1.73	0.27	0.20
GFS	RH	5	0.56	0.96	1.72	0.28	0.21
GFS	RH	6	0.56	0.95	1.71	0.27	0.19
GFS	RH	7	0.56	0.95	1.72	0.28	0.19
GFS	RH	8	0.56	1.01	1.81	0.27	0.20
GFS	RH	9	0.56	0.97	1.74	0.27	0.20
GFS	RH	10	0.56	0.99	1.78	0.28	0.20
GFS	RH	11	0.56	0.95	1.72	0.27	0.20
GFS	RH	12	0.56	0.96	1.73	0.27	0.19
GFS	RH	13	0.56	0.94	1.69	0.27	0.19
GFS	RH	14	0.56	0.98	1.76	0.27	0.19
GFS	RH	15	0.56	0.97	1.75	0.28	0.19
GFS	RH	16	0.56	0.98	1.76	0.27	0.19
GFS	RH	17	0.56	1.01	1.81	0.27	0.20
GFS	RH	18	0.56	1.04	1.87	0.27	0.20
GFS	RH	19	0.56	0.94	1.69	0.27	0.19
GFS	RH	20	0.56	0.96	1.72	0.28	0.20
GFS	RH	21	0.56	0.94	1.70	0.27	0.19
GFS	RH	22	0.56	0.94	1.69	0.27	0.19
GFS	RH	23	0.56	0.97	1.75	0.28	0.20
GFS	RH	24	0.56	0.94	1.70	0.27	0.20
GFS	RH	25	0.56	0.94	1.70	0.27	0.19
GFS	SPFH	0	0.59	0.29	0.49	0.26	0.19
GFS	SPFH	1	0.59	0.29	0.48	0.28	0.20
GFS	SPFH	2	0.59	0.33	0.55	0.28	0.21

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
GFS	SPFH	3	0.59	0.29	0.50	0.28	0.20
GFS	SPFH	4	0.59	0.29	0.49	0.26	0.19
GFS	SPFH	5	0.59	0.29	0.50	0.27	0.20
GFS	SPFH	6	0.59	0.31	0.53	0.27	0.20
GFS	SPFH	7	0.59	0.29	0.49	0.28	0.20
GFS	SPFH	8	0.59	0.33	0.57	0.28	0.22
GFS	SPFH	9	0.59	0.29	0.50	0.27	0.20
GFS	SPFH	10	0.59	0.30	0.50	0.29	0.21
GFS	SPFH	11	0.59	0.32	0.53	0.28	0.22
GFS	SPFH	12	0.59	0.33	0.56	0.28	0.21
GFS	SPFH	13	0.59	0.29	0.50	0.28	0.20
GFS	SPFH	14	0.59	0.33	0.56	0.27	0.21
GFS	SPFH	15	0.59	0.33	0.56	0.27	0.21
GFS	SPFH	16	0.59	0.30	0.51	0.28	0.21
GFS	SPFH	17	0.59	0.34	0.58	0.27	0.21
GFS	SPFH	18	0.59	0.34	0.57	0.27	0.22
GFS	**SPFH**	**19**	**0.59**	**0.28**	**0.48**	**0.29**	**0.21**
GFS	SPFH	20	0.59	0.33	0.55	0.28	0.21
GFS	SPFH	21	0.59	0.30	0.51	0.27	0.20
GFS	SPFH	22	0.59	0.28	0.48	0.27	0.21
GFS	SPFH	23	0.59	0.33	0.56	0.27	0.20
GFS	SPFH	24	0.59	0.28	0.48	0.27	0.20
GFS	SPFH	25	0.59	0.29	0.50	0.28	0.20
GFS	VVEL	0	0.33	0.40	1.23	0.29	0.25
GFS	VVEL	1	0.33	0.40	1.23	0.30	0.27
GFS	VVEL	2	0.33	0.43	1.32	0.30	0.28
GFS	VVEL	3	0.33	0.40	1.24	0.30	0.27
GFS	VVEL	4	0.33	0.41	1.25	0.29	0.27
GFS	VVEL	5	0.33	0.40	1.23	0.30	0.27
GFS	VVEL	6	0.33	0.43	1.31	0.30	0.28
GFS	VVEL	7	0.33	0.41	1.25	0.30	0.27
GFS	VVEL	8	0.33	0.43	1.33	0.31	0.28
GFS	VVEL	9	0.33	0.41	1.26	0.30	0.27
GFS	VVEL	10	0.33	0.41	1.27	0.31	0.27
GFS	VVEL	11	0.33	0.43	1.32	0.30	0.28
GFS	VVEL	12	0.33	0.43	1.33	0.30	0.27
GFS	VVEL	13	0.33	0.43	1.31	0.31	0.27
GFS	VVEL	14	0.33	0.43	1.33	0.30	0.28
GFS	VVEL	15	0.33	0.43	1.32	0.29	0.26
GFS	VVEL	16	0.33	0.43	1.31	0.30	0.27
GFS	VVEL	17	0.33	0.43	1.33	0.30	0.27

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
GFS	VVEL	18	0.33	0.44	1.33	0.30	0.28
GFS	**VVEL**	**19**	**0.33**	**0.40**	**1.23**	**0.31**	**0.27**
GFS	VVEL	20	0.33	0.43	1.32	0.30	0.28
GFS	VVEL	21	0.33	0.43	1.32	0.29	0.26
GFS	VVEL	22	0.33	0.40	1.23	0.29	0.25
GFS	VVEL	23	0.33	0.43	1.33	0.29	0.26
GFS	VVEL	24	0.33	0.40	1.23	0.29	0.26
GFS	VVEL	25	0.33	0.43	1.31	0.29	0.26
GFS	UGRD	0	0.31	0.49	1.60	0.28	0.25
GFS	UGRD	1	0.31	0.49	1.59	0.29	0.26
GFS	UGRD	2	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	3	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	4	0.31	0.49	1.59	0.30	0.28
GFS	UGRD	5	0.31	0.49	1.58	0.29	0.26
GFS	UGRD	6	0.31	0.49	1.58	0.29	0.27
GFS	UGRD	7	0.31	0.49	1.59	0.29	0.27
GFS	**UGRD**	**8**	**0.31**	**0.49**	**1.58**	**0.29**	**0.27**
GFS	UGRD	9	0.31	0.49	1.59	0.29	0.26
GFS	UGRD	10	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	11	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	12	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	13	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	14	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	15	0.31	0.49	1.59	0.30	0.27
GFS	UGRD	16	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	17	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	18	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	19	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	20	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	21	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	22	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	23	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	24	0.31	0.49	1.59	0.29	0.27
GFS	UGRD	25	0.31	0.49	1.59	0.29	0.26
GFS	VGRD	0	0.28	0.46	1.61	0.28	0.26
GFS	VGRD	1	0.28	0.46	1.61	0.29	0.26
GFS	VGRD	2	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	3	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	4	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	5	0.28	0.46	1.60	0.29	0.28
GFS	VGRD	6	0.28	0.46	1.60	0.29	0.26

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
GFS	VGRD	7	0.28	0.46	1.60	0.29	0.26
GFS	**VGRD**	**8**	**0.28**	**0.46**	**1.60**	**0.29**	**0.26**
GFS	VGRD	9	0.28	0.46	1.60	0.30	0.27
GFS	VGRD	10	0.28	0.46	1.60	0.30	0.27
GFS	VGRD	11	0.28	0.46	1.61	0.29	0.26
GFS	VGRD	12	0.28	0.46	1.61	0.29	0.27
GFS	VGRD	13	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	14	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	15	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	16	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	17	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	18	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	19	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	20	0.28	0.46	1.60	0.29	0.27
GFS	VGRD	21	0.28	0.46	1.60	0.29	0.26
GFS	VGRD	22	0.28	0.46	1.61	0.30	0.26
GFS	VGRD	23	0.28	0.46	1.61	0.30	0.27
GFS	VGRD	24	0.28	0.46	1.60	0.29	0.28
GFS	VGRD	25	0.28	0.46	1.60	0.29	0.26
GFS	ABSV	0	0.23	0.32	1.36	0.27	0.21
GFS	ABSV	1	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	2	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	3	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	4	0.23	0.32	1.35	0.29	0.22
GFS	ABSV	5	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	6	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	7	0.23	0.32	1.35	0.28	0.23
GFS	**ABSV**	**8**	**0.23**	**0.32**	**1.34**	**0.28**	**0.22**
GFS	ABSV	9	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	10	0.23	0.32	1.36	0.28	0.22
GFS	ABSV	11	0.23	0.32	1.36	0.28	0.22
GFS	ABSV	12	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	13	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	14	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	15	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	16	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	17	0.23	0.32	1.36	0.28	0.23
GFS	ABSV	18	0.23	0.32	1.37	0.28	0.22
GFS	ABSV	19	0.23	0.32	1.35	0.30	0.22
GFS	ABSV	20	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	21	0.23	0.32	1.35	0.29	0.23

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
GFS	ABSV	22	0.23	0.32	1.35	0.29	0.22
GFS	ABSV	23	0.23	0.32	1.35	0.28	0.22
GFS	ABSV	24	0.23	0.32	1.35	0.29	0.22
GFS	ABSV	25	0.23	0.32	1.35	0.28	0.22
GFS	CLWMR	0	0.31	0.88	2.81	0.29	0.20
GFS	CLWMR	1	0.31	0.85	2.70	0.31	0.21
GFS	CLWMR	2	0.31	1.03	3.28	0.30	0.23
GFS	CLWMR	3	0.31	0.87	2.78	0.30	0.21
GFS	CLWMR	4	0.31	0.84	2.69	0.30	0.21
GFS	CLWMR	5	0.31	0.92	2.94	0.29	0.22
GFS	CLWMR	6	0.31	1.04	3.30	0.31	0.22
GFS	CLWMR	7	0.31	0.90	2.85	0.32	0.22
GFS	CLWMR	8	0.31	1.04	3.31	0.30	0.22
GFS	CLWMR	9	0.31	0.92	2.92	0.30	0.21
GFS	CLWMR	10	0.31	0.88	2.79	0.30	0.21
GFS	CLWMR	11	0.31	1.02	3.24	0.31	0.22
GFS	CLWMR	12	0.31	1.02	3.26	0.30	0.22
GFS	CLWMR	13	0.31	1.01	3.20	0.30	0.22
GFS	CLWMR	14	0.31	1.04	3.30	0.30	0.22
GFS	CLWMR	15	0.31	1.04	3.30	0.31	0.22
GFS	CLWMR	16	0.31	1.03	3.27	0.30	0.22
GFS	CLWMR	17	0.31	1.05	3.33	0.30	0.22
GFS	CLWMR	18	0.31	1.04	3.31	0.30	0.22
GFS	CLWMR	19	0.31	0.82	2.61	0.30	0.20
GFS	CLWMR	20	0.31	1.04	3.31	0.30	0.22
GFS	CLWMR	21	0.31	1.01	3.21	0.30	0.23
GFS	CLWMR	22	0.31	0.82	2.60	0.30	0.20
GFS	CLWMR	23	0.31	1.02	3.25	0.30	0.22
GFS	**CLWMR**	**24**	**0.31**	**0.82**	**2.60**	**0.29**	**0.20**
GFS	CLWMR	25	0.31	1.00	3.19	0.30	0.22
NEXRAD	**BREF1**	**0**	**1.00**	**1.33**	**1.33**	**2.55**	**0.66**
NEXRAD	BREF1	1	1.00	1.57	1.57	2.74	0.68
NEXRAD	BREF1	2	1.00	1.60	1.60	2.69	0.69
NEXRAD	BREF1	3	1.00	1.54	1.54	2.73	0.68
NEXRAD	BREF1	4	1.00	1.44	1.44	2.67	0.69
NEXRAD	BREF1	5	1.00	1.52	1.52	2.60	0.68
NEXRAD	BREF1	6	1.00	1.54	1.54	2.91	0.98
NEXRAD	BREF1	7	1.00	1.54	1.54	2.63	0.70
NEXRAD	BREF1	8	1.00	1.58	1.58	2.66	0.70
NEXRAD	BREF1	9	1.00	1.54	1.54	2.62	0.68
NEXRAD	BREF1	10	1.00	1.50	1.50	2.62	0.69

... continued on next page

... continued from previous page

DSN	**Var**	**Wnum**	r_{orig}	r_{new}	r_{new}/r_{orig}	t_{enc}	t_{dec}
NEXRAD	BREF1	11	1.00	1.46	1.46	2.72	0.69
NEXRAD	BREF1	12	1.00	1.43	1.43	2.75	0.69
NEXRAD	BREF1	13	1.00	1.42	1.42	2.73	0.69
NEXRAD	BREF1	14	1.00	1.49	1.49	2.71	0.68
NEXRAD	BREF1	15	1.00	1.55	1.55	2.69	0.69
NEXRAD	BREF1	16	1.00	1.54	1.54	2.71	0.70
NEXRAD	BREF1	17	1.00	1.71	1.71	2.67	0.68
NEXRAD	BREF1	18	1.00	1.71	1.71	2.66	0.70
NEXRAD	BREF1	19	1.00	1.42	1.42	2.68	0.69
NEXRAD	BREF1	20	1.00	1.53	1.53	2.64	0.70
NEXRAD	BREF1	21	1.00	1.41	1.41	2.58	0.69
NEXRAD	BREF1	22	1.00	1.43	1.43	2.73	0.69
NEXRAD	BREF1	23	1.00	1.43	1.43	2.73	0.69
NEXRAD	BREF1	24	1.00	1.41	1.41	2.67	0.68
NEXRAD	BREF1	25	1.00	1.41	1.41	2.60	0.69

For each variable in the table above, the Wnum=0 test shows the minimum, or very close to the minimum, value of $\boldsymbol{r_{new}}$, which implies the minimum of $\boldsymbol{r_{new}/r_{orig}}$.

This implies we can standardize on wavelet 0, and omit implementing the other wavelets.

7 Conclusions

As shown in section 6.2, the run length encoding and and Lempel-Ziv compression steps add little if any benefit to the compression pipeline. In addition the Lempel Ziv compression significantly increases the encoding and decoding times. So we can omit the run length encoding and Lempel-Ziv compression from future work.

Similarly, as shown in section 6.3, in general the wavelet 0 performs the best and there is no reason to add the complexity of multiple wavelets. So we can standardize on wavelet 0 for future work.

There are a few reasons why the current compression system, Sengcom, may not compress as well as Grib2.

- **No IP software**. Grib2 uses JPEG2000, which makes extensive use of patented software. The patent holders have agreed not to enforce claims against “conforming” JPEG2000 implementations. Unfortunately that does not include Sengcom.
- **We’re using 1 dimensional wavelets**. The decision was to reduce the implementation complexity by having the scanner first in the pipeline. That way all subsequent stages in the pipeline need deal only with 1-dimensional data. Using multidimensional wavelets instead of 1 dimensional might improve the compression; but it would certainly increase the implementation complexity.
- **Strict control over max absolute error**. Many wavelet systems based on mean error or other less stringent error measures might obtain better compression, but without the tight control on max absolute error.

There are at least two possible paths forward.

1. Continue working with Sengcom to improve the compression. Some possibilities are:
    (a) Try using n-dimensional wavelet transforms, by moving the scanning after the wavelet transform in the pipeline. This might improve the compression, but would require a significant increase in software complexity.
    (b) Although the current scanner seems to function well, there may be refinements in the scanning order that would help.
2. Abandon Sengcom and attempt to implement the entire JPEG2000 standard using 31 bit precision, as opposed to the 16 bit precision of existing implementations.
   This would have to be a “conforming” implementation of JPEG2000, in order to satisfy the IP patent issues. Possibly, this would result in compression closer to that of Grib2. The JPEG2000 specification is large, complex, evolving, and in some cases not well defined. This would be a large effort.

A Appendix: Partial list of existing compression systems

AI, in Adobe Illustrator

- **License:** proprietary

BEF, by Unified Color [47]

- **License:** proprietary

B44 (used in OpenEXR [22])

- **Algorithm:** Lossy 16 bit or lossless uncompressed 32 bit

B44A (used in OpenEXR [22])

- **Algorithm:** Lossy 16 bit
- **Lossless uncompressed 32 bit**

CCSDS

- **Lossy 12 bit**
- **Algorithm:** Simple differencing followed by an entropy coder.
- **References**
  - `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1559719&tag=1` The new CCSDS image compression recommendation
  - `http://en.wikipedia.org/wiki/CCSDS_122.0-B-1`
    Discusses “Known flaws”. In the encoder’s current form an image pixel dynamic range of up to 16 bits is supported, whereas ICER and JPEG2000 support 24 and 32 bitplanes.

CDR: Corel Draw

- **License:** proprietary

OpenEXR: OpenEXR [22]

- **License:** similar to BSD
- **Origin:** Developed by Industrial Light and Magic and Pixar.
  Now at `http://www.openexr.com`
- **Bits:** 16, 32
- **Loss:** Lossless or lossy

- **Algorithm:** EXR is more of an encapsulation, and uses a variety of compression methods inside: RLE, Zip, PIZ, PXR24, B44, B44A. It is the standard file format used by Industrial Light and Magic.
- **Implementations:**
  - `http://www.openexr.com`
- **References:**
  - `http://www.openexr.com/`
  - `http://en.wikipedia.org/wiki/OpenEXR`

ECW - Enhanced Compression Wavelet [20]

- **License:** proprietary, by Earth Resource Mapping, now owned by ERDAS, which is owned by Intergraph. Has patents.
- **Origin:** Developed for satellite imagery for hyperspectral data, using embedded processors. Essentially ECW was replaced by JPEG2000.
- **Bits:**
- **Loss:** Lossy
- **Algorithm:** Discrete wavelet transform. Fast, low memory.

ICER [23]

- **Origin:** Used for all image transmissions from the NASA Mars Rover program. Similar to JPEG2000. Designed for simple, low-end processors without floating point arithmetic.
- **Bits:** 12
- **License:** proprietary, patented
- **Implementations:** none available
- **References:**
  - `http://en.wikipedia.org/wiki/ICER`

JPEG [29]

- **License:** Open
- **Bits:** 8 or 12
- **Patent claims** `http://www.jpeg.org/newsrel1.html`

JPEG 2000 [2][21]

- **License:** Open
- **Bits:** 8 or 16 per channel. Some posts have claimed that JPEG2000 supports 32 bits. It does not - I've tested it. Even when declaring the number of bits as 32, when the input values exceed 35,000 in some cases the output values are silently corrupted. There is no indication of error.

- **Loss:** Lossless or lossy
- **Algorithm:** Divide into tiles. Then integer or floating point lifting wavelets.
- **Implementations:**
  - C: `http://en.wikipedia.org/wiki/JasPer`
  - C: `http://www.openjpeg.org/`
  - Java: `http://code.google.com/p/jj2000/`
- **References:**
  - `http://www.jpeg.org/jpeg2000/`
  - `http://www.openjpeg.org/`
  - `http://code.google.com/p/jj2000/`
  - `http://en.wikipedia.org/wiki/JPEG_2000`

JPEG LS, Lossless JPEG [29]

- **Origin:** Predecessor to JPEG2000
- **References:**
  - `http://en.wikipedia.org/wiki/Lossless_JPEG`

JPEG XR, HD Photo

- **License:** Ambiguous. The algorithm may be open; the MS implementation (HD Photo) appears to be proprietary.
- **Bits:** up to 32
- **Loss:** Lossless or lossy
- **Algorithm:** Divide into tiles, then 16x16 blocks, then 4x4 blocks. Then integer lifting wavelets. Then multiple entropy encoders for the various streams of coefficients. Reports say the adaptive entropy coding is complex.
- **Implementations:**
  - Microsoft HD Photo, in C: `http://www.itu.int/rec/T-REC-T.835`
- **References:**
  - `http://en.wikipedia.org/wiki/JPEG_XR`
  - `http://www.barrypearson.co.uk/articles/hdp/`
  - `http://channel9.msdn.com/Forums/Coffeehouse/The-current-state-of-Jpeg-XR`

PIZ (used in EXR)

- **Algorithm** Haar wavelet with Huffman encoding.
- **References:**
  - `http://www.mail-archive.com/openexr-user@nongnu.org/msg00258.html`
    On Wed, 27 Feb 2008, Florian Kainz wrote:

  Unfortunately there isn't a document that describes PIZ (or any of the other compression methods) in detail. If anyone wants to write such a description, I'd be happy to put it on the web site, along with the other documentation.
  - `http://lists.nongnu.org/archive/html/openexr-user/2005-01/msg00004.html`
    From: Florian Kainz
    Subject: Re: [Openexr-user] PIZ compression
    Date: Wed, 19 Jan 2005 17:46:27 -0800
    However, I doubt that PIZ would be very good for general-purpose data compression. PIZ converts the floating-point (half) pixel data to integers, applies a Haar Wavelet transform to the integer data, and Huffman- encodes the output. Conversion from floating-point to integer and the wavelet transform are designed to alter the statistical distribution of the pixel values so that small numbers occur much more frequently than large ones. Both steps were specifically designed for photographic images, and they tend to work well for this purpose, but there is no reason to believe that they would do anything useful for other kinds of data, for example, text. The Huffman coder directly encodes 16-bit symbols rather than individual bytes. The coder would work with with text or similar data, but the compression ratios that can be achieved by Huffman- encoding individual characters without any preprocessing tend to be unimpressive. Overall, I think you are better off using zlib/gzip/bzip2 for general-purpose compression.
    Florian
- `http://www.openexr.com/TechnicalIntroduction.pdf`

PNG

- **Bits:** 8.

PXR24 (used in EXR)

- **License:** Open, probably
- **Origin:** Developed in Pixar for EXR.
- **Bits:** 24.
- **Algorithm:** Round 32 bits to 24, then lossless zlib.
- **Implementations:**
  - See EXR.
- **References:**
  - `http://www.openexr.com/`
  - www.openexr.com/TechnicalIntroduction.pdf
  - `http://en.wikipedia.org/wiki/OpenEXR`

WebP

- **License:** Open

- **Origin:** Google
- **Bits:** 8.
- **Algorithm:** Adjacent block prediction, then 4x4 blocks, then lossless integer discrete cosine transform.
- **Implementations:**
  - `https://developers.google.com/speed/webp/`
- **References:**
  - `http://en.wikipedia.org/wiki/Webp`
  - `https://developers.google.com/speed/webp/`

x264 (H.264/MPEG-4 AVC)

- **Origin:** High definition video compression. It is one of the most common formats used with Blu-ray discs.
- **Algorithm:** Large, complex spec. Variable block size motion compensation, spatial transforms like DCT on 4x4 or 8x8 regions, quantization with log step sizes, multiple entropy coding systems, etc.
- **Patents:** Patents and licensing fees are enforced.
- **References:**
  - `http://www.videolan.org/developers/x264.html`
  - `http://en.wikipedia.org/wiki/X264`

References

[1] M. D. Adams and F. Kossentini. Reversible integer-to-integer wavelet transforms for image compression: Performance evaluation and analysis. *IEEE Trans. Image Processing*, 9(6):1010–1024, June 2000.

[2] Michael D. Adams. The JPEG-2000 still image compression standard, July 01 2000.

[3] H. F. Ates and M. T. Orchard. Spherical coding algorithm for wavelet image compression. *IEEE Trans. Image Processing*, 18(5):1015–1024, May 2009.

[4] Ali Bilgin, George Zweig, Michael W. Marcellin, and Ali Bilgin. Three-dimensional image compression with integer wavelet transforms, 2000.

[5] Joseph B. Boettcher, Qian Du, and James E. Fowler. Hyperspectral image compression with the 3D dual-tree wavelet transform. In *IGARSS*, pages 1033–1036. IEEE, 2007.

[6] N. V. Boulgouris, D. Tzovaras, and M. G. Strintzis. Lossless image compression based on optimal prediction, adaptive lifting, and conditional arithmetic coding. *IEEE Trans. Image Processing*, 10(1):1–14, January 2001.

[7] M. Burtscher and P. Ratanaworabhan. FPC: A high-speed compressor for double-precision floating-point data. *IEEE Trans. on Computers*, 58(1):18–31, January 2009.

[8] Weiting Cai and Malek Adjouadi. Minimization of boundary artifacts on scalable image compression using symmetric-extended wavelet transform. In *ITCC (1)*, pages 598–602. IEEE Computer Society, 2004.

[9] A. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo. Lossless image compresion using integer to integer wavelet transforms. pages 596–599, 1997.

[10] R. Calderbank, Ingrid Daubechies, Wim Sweldens, and Boon-Lock Yeo. Wavelet transforms that map integers to integers. Technical report, Department of Mathematics, Princeton University, 1996.

[11] Guoxiang Chen and Peter B. Harrington. Real-time two-dimensional wavelet compression and its application to real-time modeling of ion mobility data. *Analytica Chimica Acta*, 490(1-2):59–69, August 2003.

[12] J. L. Chen and J. Yang. Improved image coding algorithm based on embedded zerotree wavelet. In *International Congress on Image and Signal Processing*, pages 1–4, 2009.

[13] R. L. Claypoole, G. M. Davis, W. Sweldens, and R. G. Baraniuk. Nonlinear wavelet transforms for image coding via lifting. *IEEE Trans. Image Processing*, 12(12):1449–1459, December 2003.

[14] Ingrid Daubechies and Wim Sweldens. Factoring wavelet transforms into lifting steps. Technical report, Bell Laboratories, Lucent Technologies, 1996.

[15] Shilpa S. Dhulap and Sanjay L. Nalbalwar. Image compression based on IWT, IWPT & DPCM-IWPT. 2010.

[16] Wesley Ebisuzai. wgrib2 for GRIB-2, version 1.9.1.c, April 2011.

[17] M. Grangetto, E. Magli, M. Martina, and G. Olmo. Optimization and implementation of the integer wavelet transform for image coding. *IEEE Trans. Image Processing*, 11(6):596–604, June 2002.

[18] HDF Group. SZIP compression in HDF products.

[19] HDF Group. HDF5 home page, April 2011.

[20] Intergraph. ECW format, July 2011.

[21] JPEG Committee. The ICER progressive wavelet image compressor, July 2011.

[22] Florian Kainz and Rod Bogart. Technical introduction to OpenEXR, February 2009.

[23] A. Kiely and M. Klimesh. The ICER progressive wavelet image compressor, November 2003.

[24] Rade Kutil. Zerotree image compression using anisotropic wavelet packet transform. In Touradj Ebrahimi and Thomas Sikora, editors, *Visual Communications and Image Processing 2003*, volume 5150 of *Proceedings of SPIE*, pages 1417–1427. SPIE, 2003.

[25] K. Kuzume and K. Niijima. Design of optimal lifting wavelet filters for data compression. In *Time-Frequency and Time-Scale Analysis, 1998, Proceedings of the IEEE-SP International Symposium on*, pages 337–340, 1998.

[26] Jianyu Lin and Mark Smith. New perspectives and improvements on the symmetric extension filter bank for subband/wavelet image compression. *IEEE Transactions on Image Processing*, 17(2):177–189, 2008.

[27] Anna Linderhed. *Adaptive Image Compression With Wavelet Packets And Empirical Mode Decomposition*. PhD thesis, Linkping University, 2004.

[28] Anna Linderhed. Compression by image empirical mode decomposition. In *ICIP*, pages I: 553–556, 2005.

[29] William B. Pennebaker and Joan L. Mitchell. *JPEG Still Image Data Compression Standard, 3rd ed.* springer, 1993.

[30] Stephen Pollock and Iolanda Lo Cascio. Orthogonality conditions for non-dyadic wavelet analysis. Technical Report 529, Dept of Economics, Queen Mary University of London, 2005.

[31] Stephen Pollock and Iolanda Lo Cascio. Non-dyadic wavelet analysis. In Erricos John Kontoghiorghes and Cristian Gatu, editors, *Optimisation, Econometric and Financial Analysis*, volume 9 of *Advances in Computational Management Science*, pages 167–203. Springer Verlag, 2007.

[32] S.P. Raja and A. Suruliandi. Analysis of efficient wavelet based image compression techniques. In *2010 Second International conference on Computing, Communication and Networking Technologies*, 2010.

[33] J. Reichel, G. Menegaz, M. J. Nadenau, and M. Kunt. Integer wavelet transform for embedded lossy to lossless image compression. *IEEE Trans. Image Processing*, 10(3):383–392, March 2001.

[34] A. Said and W. A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. Circuits and Systems for Video Technology*, 6(3):243–250, 1996.

[35] Edward L. Schwartz, Kathrin Berkner, and Michael J. Gormish. Optimal tile boundary artifact removal with CREW, 1999.

[36] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, jul,oct 1948.

[37] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. Signal Processing*, 41(12):3445–3462, 1993.

[38] F. Sheng, A. Bilgin, P. Sementilli, and M. Marcellin. Lossy and lossless image compression using reversible integer wavelet transforms. In *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, pages 876–880, 1998.

[39] John R. Smith and Shih fu Chang. Space adaptive wavelet packet image compression, 2007.

[40] Tilo Strutz. Design of three-channel filter banks for lossless image compression. In *ICIP*, pages 2841–2844. IEEE, 2009.

[41] Tilo Strutz. Lifting parameterisation of the 9/7 wavelet filterbank and its application in lossless image compression. In *ISPRA 09 Proceedings of the 8th WSEAS international conference on Signal processing, robotics and automation*, pages 161–166. WSEAS Press, February 2009.

[42] Tilo Strutz. Wavelet filter design based on the lifting scheme and its application in lossless image compression. *WSEAS Transactions on Signal Processing*, 5(2):53–62, feb 2009.

[43] Tilo Strutz and Ines Rennert. Two-dimensional integer wavelet transform with reduced influence of rounding operations. *EURASIP Journal on Advances in Signal Processing*, 12(75), 2012.

[44] Xiaoli Tang and William A. Pearlman. Lossy-to-lossless block-based compression of hyperspectral volumetric data. In *ICIP*, pages 3283–3286, 2004.

[45] Unidata. Unidata home page, April 2011.

[46] Unidata. Unidata spring 2012 metrics data, 2012.

[47] Unified Color Technologies. BEF format, July 2012.

[48] Bryan E. Usevitch. A tutorial on modern lossy wavelet image compression: Foundations of JPEG 2000, 2001.

[49] Geert Uytterhoeven, Dirk Roose, and Adhemar Bultheel. Wavelet transforms using the lifting scheme, 1997.

[50] John D. Villasenor, Benjamin Belzer, and Judy Liao. Wavelet filter evaluation for image compression. *IEEE Transactions on Image Processing*, 4(8):1053–1060, August 1995.

[51] James S. Walker. *The Transform and Data Compression Handbook*, chapter Wavelet-based Image Compression. CRC Press, Boca Raton, FL, 2001.

[52] Lei Wang, Jiaji Wu, Licheng Jiao, and Guangming Shi. 3D medical image compression based on multiplierless low-complexity RKLT and shape-adaptive wavelet transform. In *ICIP*, pages 2521–2524. IEEE, 2009.

[53] Gaofeng Wei, Hongxu Jiang, and Rui Yang. Linear-regression model based wavelet filter evaluation for image compression. In *APWCS*, pages 315–318. IEEE Computer Society, 2010.

[54] Mladen Victor Wickerhauser. Designing a custom wavelet packet image compression scheme, with applications to fingerprints and seismic data, April 23 1999.

[55] J. R. Williams and K. Amaratunga. A discrete wavelet transform without edge effects using wavelet extrapolation. Technical report, Intelligent Engineering Systems Laboratory, MIT, 1995.

[56] World Meteorological Organization. WMO International Codes, April 2011.

[57] World Meteorological Organization. NCEP WMO GRIB2 documentation, 2012.

[58] R. Q. Xiong, J. Z. Xu, and F. Wu. A lifting-based wavelet transform supporting non-dyadic spatial scalability. In *ICIP*, pages 1861–1864, 2006.

[59] Zixiang Xiong, Xiaolin Wu, Samuel Cheng, and Jianping Hua. Lossy-to-lossless compression of medical volumetric data using three-dimensional integer wavelet transforms. *IEEE Trans. Med. Imaging*, 22(3):459–470, 2003.

[60] Jing-jing Zheng, Jin-yun Fang, and Cheng-de Han. Reversible integer wavelet evaluation for DEM progressive compression. In *IGARSS*, pages 52–55. IEEE, 2009.

[61] Jacob Ziv and Abraham Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343, May 1977.