

Lossy Compression of ACS images

Colin Cox,
January 20, 2004

ABSTRACT

A method of compressing images stored as floating point arrays was proposed several years ago by White and Greenfield. With the increased image sizes encountered in the last few years and the consequent need to distribute large data volumes, the value of applying such a procedure has become more evident. Methods such as this which offer significant compression ratios are lossy and there is always some concern that statistically important information might be discarded. Several astronomical images have been analyzed and, in the examples tested, compression ratios of about six were obtained with no significant information loss.

Introduction

With the large volumes of data that STScI processes it becomes of interest to deal with large images in compressed form. This is of particular value for the delivery of images to external computers where ultimately there is a limit to the available transmission rate. Lossless compression methods on complex images only offer a reduction in volume of 25% to 50% and sometimes even increases the size of the final files. Lossy methods can frequently compress by factors of ten or more, but there is always concern about what is lost. If images are designed only for viewing, compression factors greater than ten can be achieved with no visible difference between an original and a compressed image. For more critical measurement applications such as astrometry and photometry, one is more concerned that mathematical properties be retained. The question is, has anything scientifically important been lost by the compression.

A compression method has been developed at STScI that specifically addresses the problem of dealing with images expressed as floating point numbers. A description of the method is given in White and Greenfield, 1999 and only a few pertinent features and the performance on several test images will be described here.

Consideration is being given to delivering images in compressed form. We would supply alongside, the program which performs the decompression. The same executable is able to perform the compression, so users can store their own FITS data in compressed form and reconstruct it for analysis. A user would always have the option of receiving uncompressed data. A longer term objective is to create analysis programs which operate directly on the compressed files.

Method Outline

The first stage of the algorithm converts the floating point numbers into integer form. Normally the floating point representation is much finer than needed for the application; many of the bits are not significant. For example, raw ACS data start as 16-bit integers and in CALACS processing are mapped into 32-bit real values which incorporate 24 bits in the mantissa. Retaining non-significant bits allows freedom from worry about rounding error possibly corrupting data, but restricts the compressibility. So the first step is to decide how small the step between successive number representations should be.

Any physical observation will contain some noise, partly statistical and probably some instrumental. Variations in the true signal that are much smaller than the noise level cannot really be measured and so, at some point, we will not lose information if we do not record these small variations. The White and Greenfield algorithm takes a section of the image and first calculates the difference between successive pixels. The median value of this difference distribution multiplied by 1.483 is used as an estimate of the variance of the noise level. The factor is derived by analogy to a Gaussian distribution with the same median value. This is clearly a good estimate if most of the image area is sky background or if the signal varies slowly across the field. The image is then converted to a scaled integer array so that a selectable number of bits represents the size of this variance.

This is the only step which permits any information loss. The successive operations use the Rice algorithm to losslessly compress the integer array and write the result into a FITS table. The original description of the algorithm is given in a Nov 1991 JPL publication but a more easily found source is given in a web page in the references.

Test Images

White and Greenfield describe several tests involving astrometry and photometry on WFPC2 data plus a simulation involving averaging a set of spectra with a low signal-to-noise ratio. In all cases the compression was found to have a negligible effect on the analysis.

I have chosen several ACS images which are typical observations and should test the limits of the algorithm. Two are crowded stellar fields, one is an image with large areas not being background and the fourth contains many non-stellar objects. The first test was to compress each image, uncompress it and compare the original and reconstituted image. Absolutely no differences were visible even when images were blinked. The next test was to subtract the raw and final images and see what the difference magnitude was and look for any structure. Where an error image was available for an image, the ratio of the difference to the error was studied. If the differences are small compared with the errors, then we are not losing meaningful data by these differences. A more stringent test, which is more closely related to how the data will be used, was to run the program **daofind** on original and reconstituted images. **daofind** locates the position of each star using fitting to selectable PSF shapes and calculates a magnitude. In these studies no attempt was made to find the proper zero point; only the comparative results were considered. A third type of test was to use the utility **sextractor** which identifies such things as galaxies and irregularly shaped objects, again using mathematical properties of the image which could be degraded by the compression procedure.

Stellar fields

The first image is of a section of 47 Tucanae imaged with the HRC. The image is shown in Figure 1. The reconstructed image could be shown for comparison, but visually it is indistinguishable from the original even on a monitor when one can zoom in and manipulate the stretch and contrast. Figure 2 shows the difference image in which can be seen faint streaks. This arises because each line of the image is treated separately and so can have slightly different levels. Figure 3 shows a histogram of the difference image while Figure 4 shows a histogram of the difference divided by the error of the original. The rectangular distribution is to be expected from the first stage of the compression which is essentially a rounding. The difference will just be the shift from a general real number to the nearest scaled integer. The distribution would be more exactly rectangular for a single compressed section. The different sections do have slightly different quantization steps. As can be seen, the differences are always less than a tenth of the original error size. Figure 5 shows a WFC image which has gone through a drizzling procedure which includes the removal of cosmic rays. Again there is no visible or functional difference between this and the reconstructed image. **daofind** found the same 2882 stars in each image with the same centroids and magnitudes.

Figure 1: An HRC image of 47 Tucanae

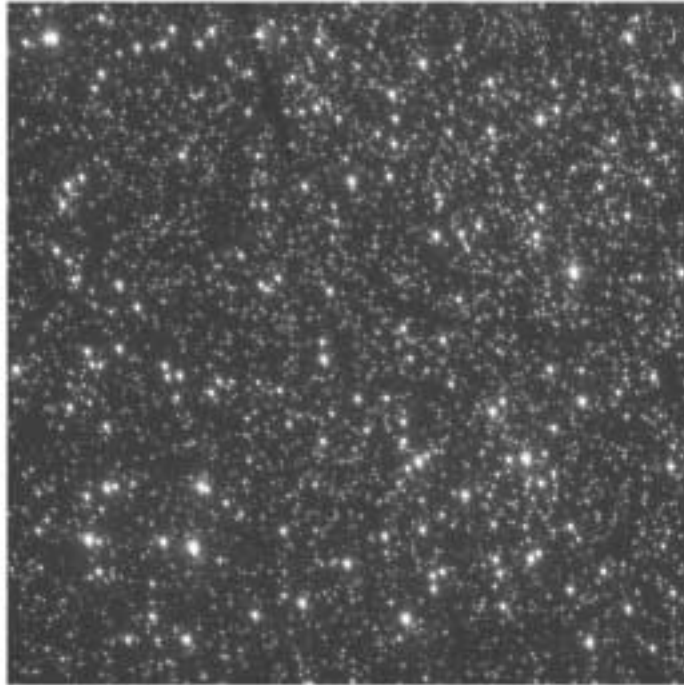


Figure 2: Difference between original and reconstituted image

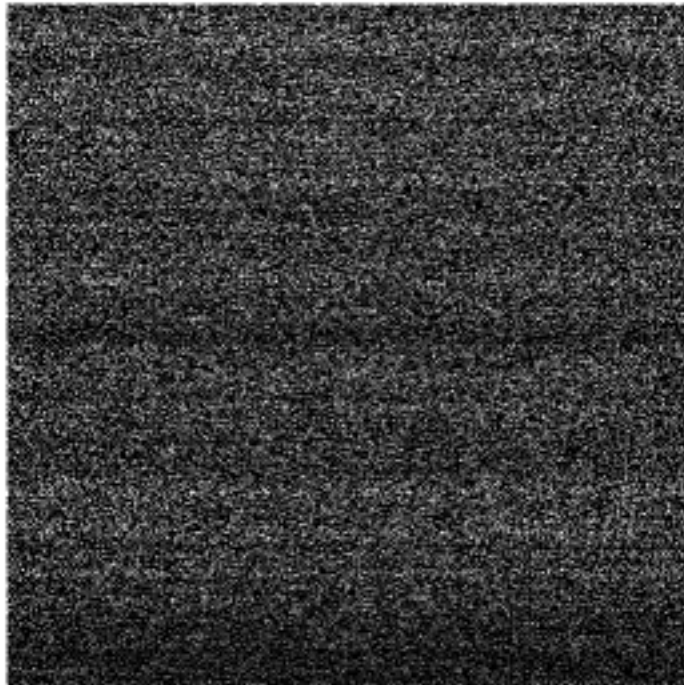


Figure 3: Histograms of difference image

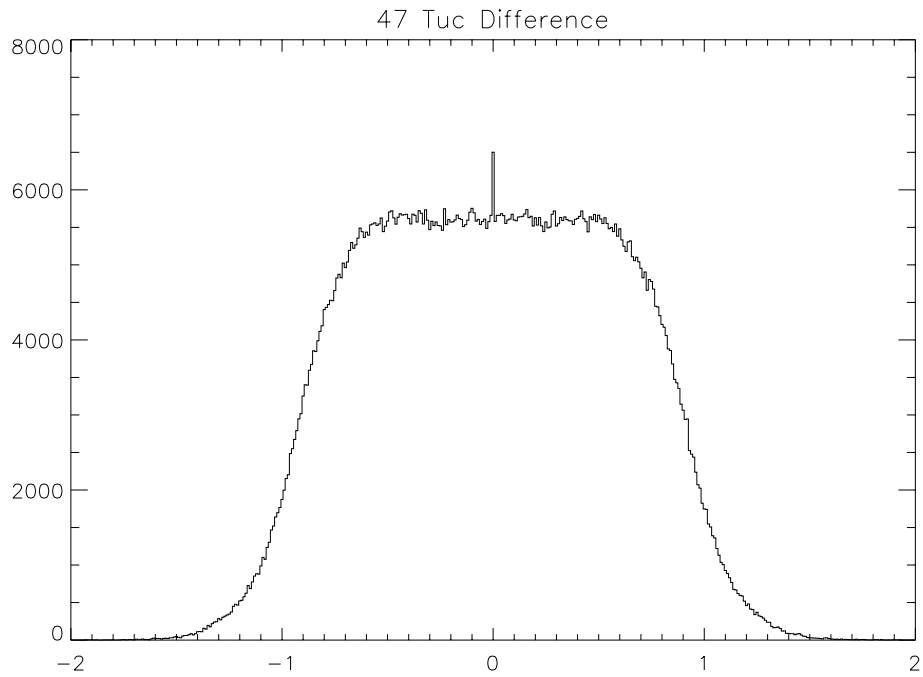


Figure 4: Histogram of ratio image

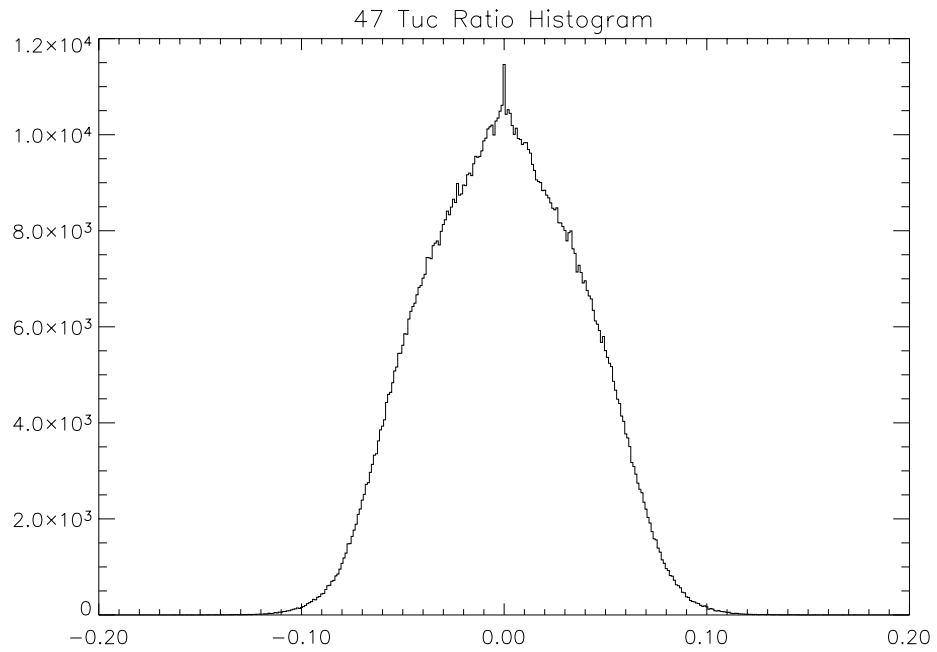
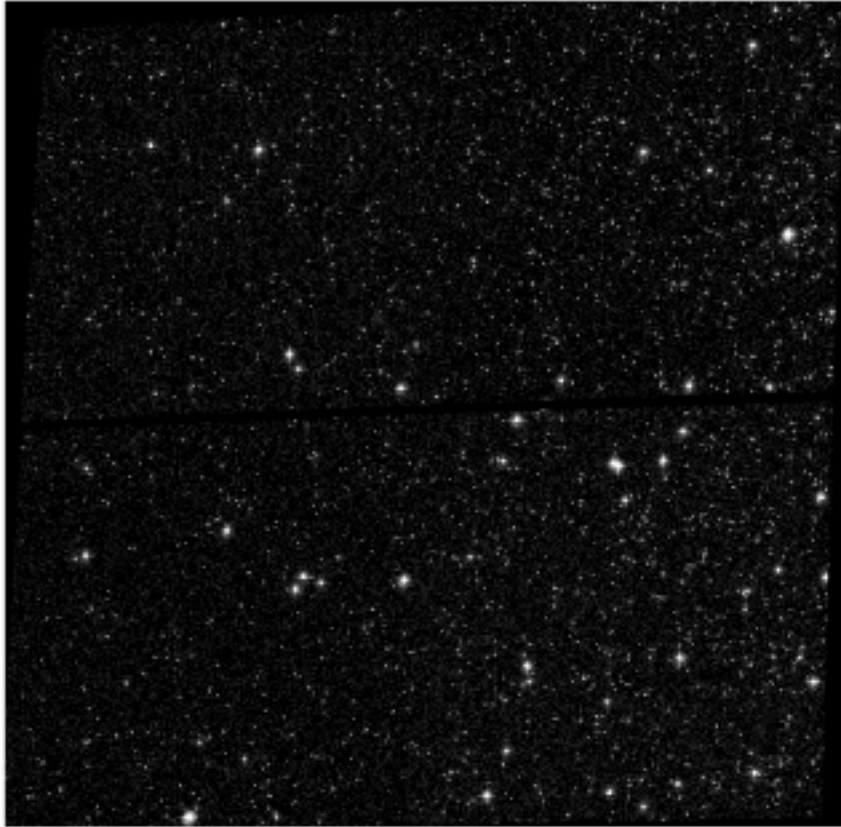


Figure 5: ACS/WFC Image



A section of the output summary from **daofind** is given for each image in the following tables.

XCENTER	YCENTER	MAG	SHARP	SROUND	GROUND	ID
3073.693	2130.559	-1.429	0.645	-0.174	0.039	1501
3467.698	2131.352	-1.248	0.624	-0.145	0.024	1502
761.542	2131.869	-1.451	0.916	-0.355	-0.341	1503
3822.269	2132.088	-3.667	0.705	0.191	0.150	1504
3011.271	2139.097	-2.408	0.748	0.122	-0.065	1505
3628.487	2146.299	-2.030	0.685	0.305	-0.068	1506
3058.853	2146.628	-0.050	0.873	-0.300	-0.561	1507
1429.107	2150.769	-2.378	0.695	0.412	0.234	1508
1189.949	2158.268	-2.435	0.629	0.256	0.041	1509
414.171	2158.919	-2.649	0.670	0.387	0.226	1510
1693.390	2162.829	-1.127	0.586	0.258	-0.165	1511
639.247	2169.317	-0.759	0.732	0.465	0.073	1512
2084.321	2174.465	-2.377	0.701	0.654	0.126	1513
939.498	2186.435	-2.075	0.589	0.743	0.091	1514
569.180	2191.927	-0.427	0.648	0.291	0.064	1515
975.938	2194.536	-0.909	0.950	0.259	0.418	1516
2870.752	2195.066	-3.415	0.539	0.311	0.424	1517
2151.067	2205.584	-4.225	0.657	0.488	0.278	1518
843.221	2207.317	-1.037	0.762	0.551	0.047	1519
1239.034	2209.130	-2.461	0.685	0.264	-0.084	1520

Table 1. Sample of daofind output derived from standard image

XCENTER	YCENTER	MAG	SHARP	SROUND	GROUND	ID
3073.693	2130.558	-1.429	0.645	-0.174	0.039	1501
3467.698	2131.352	-1.247	0.624	-0.144	0.025	1502
761.543	2131.869	-1.451	0.915	-0.354	-0.341	1503
3822.269	2132.088	-3.667	0.705	0.191	0.150	1504
3011.271	2139.097	-2.408	0.748	0.122	-0.066	1505
3628.486	2146.300	-2.030	0.685	0.305	-0.068	1506
3058.852	2146.628	-0.049	0.873	-0.301	-0.561	1507
1429.107	2150.769	-2.378	0.695	0.412	0.234	1508
1189.949	2158.268	-2.435	0.629	0.256	0.041	1509
414.171	2158.920	-2.649	0.670	0.387	0.226	1510
1693.390	2162.829	-1.127	0.586	0.258	-0.165	1511
639.248	2169.316	-0.759	0.732	0.465	0.073	1512
2084.321	2174.465	-2.377	0.701	0.654	0.125	1513
939.498	2186.435	-2.075	0.589	0.744	0.091	1514
569.181	2191.927	-0.426	0.648	0.289	0.063	1515
975.938	2194.537	-0.909	0.950	0.260	0.418	1516
2870.752	2195.066	-3.415	0.539	0.311	0.424	1517
2151.067	2205.584	-4.225	0.657	0.488	0.278	1518
843.221	2207.317	-1.038	0.762	0.551	0.047	1519
1239.034	2209.130	-2.461	0.685	0.265	-0.084	1520

Table 2. Matching sample of daofind output from reconstructed image

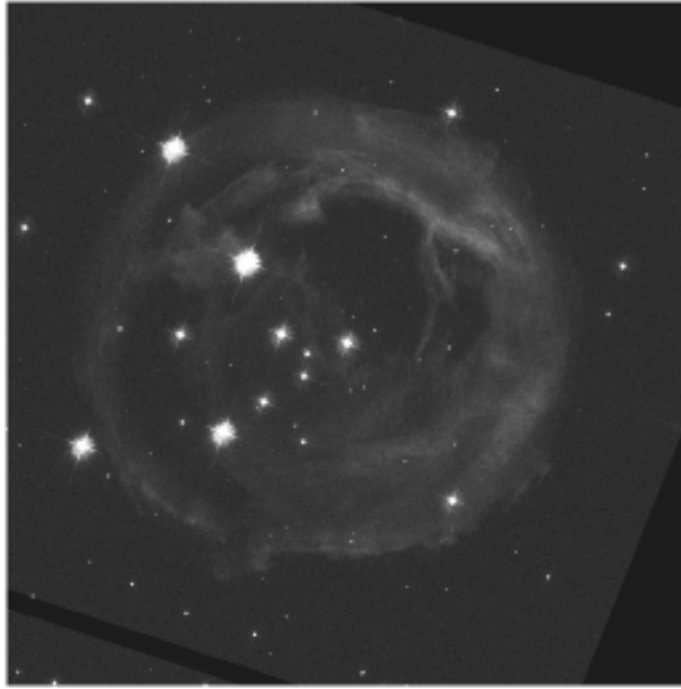
Occasional differences between matching positions and magnitudes are seen only in the third decimal place.

The image of the V838 Monoceros light echo shell, Figure 6, is an example where most of the image contains source illumination and not much is sky background. Nevertheless, the compression works just as well, again causing no visible change in the image and differences much smaller than the signal.

Finally, a sample image from the Great Observatories Origins Deep Survey (GOODS) containing a number of distant irregular galaxies was used in conjunction with a source extractor (**Sextractor**) program. This program generates a catalog of stars and galaxies found in the image along with several descriptive parameters. The catalogs generated from the image shown in Figure 8, before and after the compression/decompression sequence

were compared. This turns out to be a sterner test of the compression and does result in some differences.

Figure 6: Light Echo Image



The program hunts for faint objects at the limit of the useful signal-to-noise ratio and is therefore sensitive to modifications of the noise spectrum caused by the compression and decompression. 1853 matching sources were located in the original and reconstructed images. 29 were found in the original but missed in the reconstructed image while 39 unmatched sources were found in the reconstructed image. The question that must be addressed is, would using the reconstructed image cause any real source to be overlooked as compared with studying the original image? Figure 9 shows a magnified portion of the image with the sources detected in the original image marked by the best fit ellipses. The mean radius of the contour is drawn at twice the full width at half maximum position. Many of these do not appear to be real sources. The additional circular markers indicate which of the marked sources are not detected in the reconstructed image. For clarity, Figure 10 shows the same area with just the second set of markers. Even with the scale set to histogram equalization and the stretch set to maximize the visibility of faint objects, it is not clear that there is really a source at the center of these regions.

The catalog output by SExtractor gives the total count of the fitted elliptical source plus an error estimate. Dividing the count by the error provides a sort of signal to noise statistic. The distribution of this statistic as a function of detected magnitude is shown in Figure 7. The objects missed in the reconstructed image are shown as diamond shapes and cluster around 28th magnitude, near the limit of detection for this image.

Figure 7: Ratio of flux to flux error as a function of magnitude

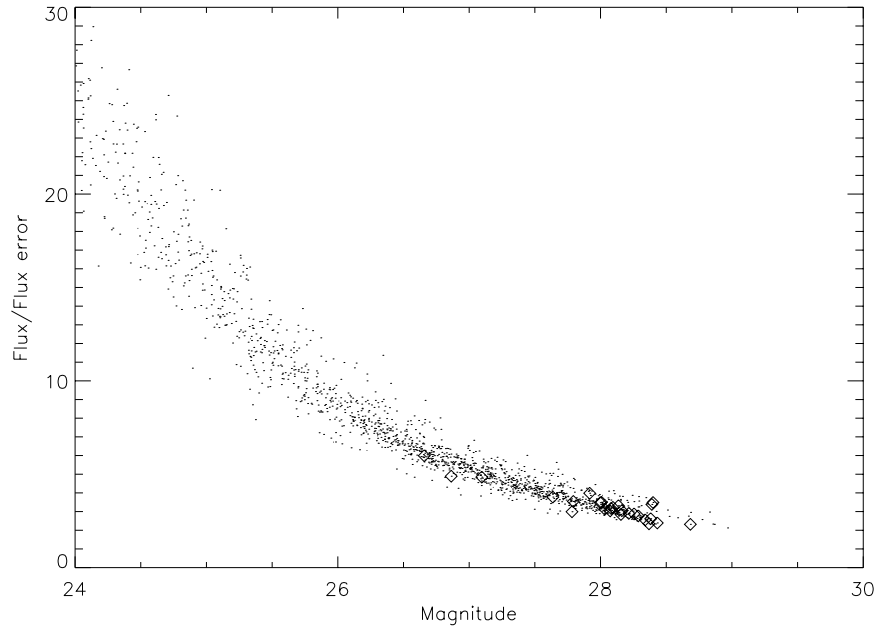


Figure 8: Section of a GOODS image showing non-stellar objects



Figure 9: Objects found by Source extractor

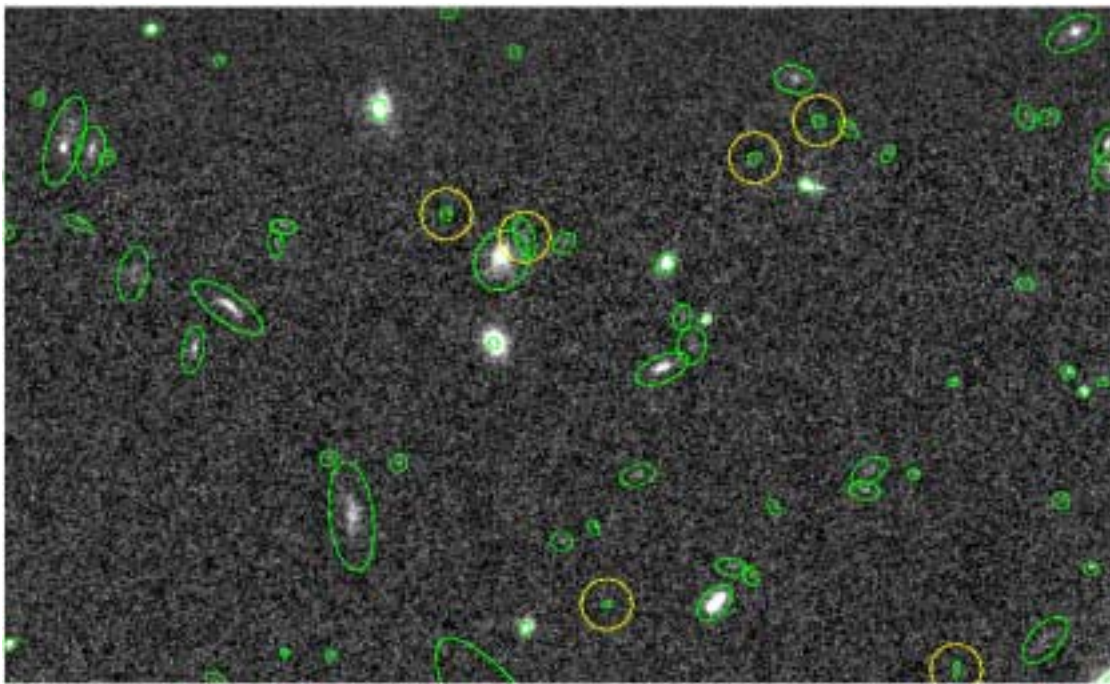
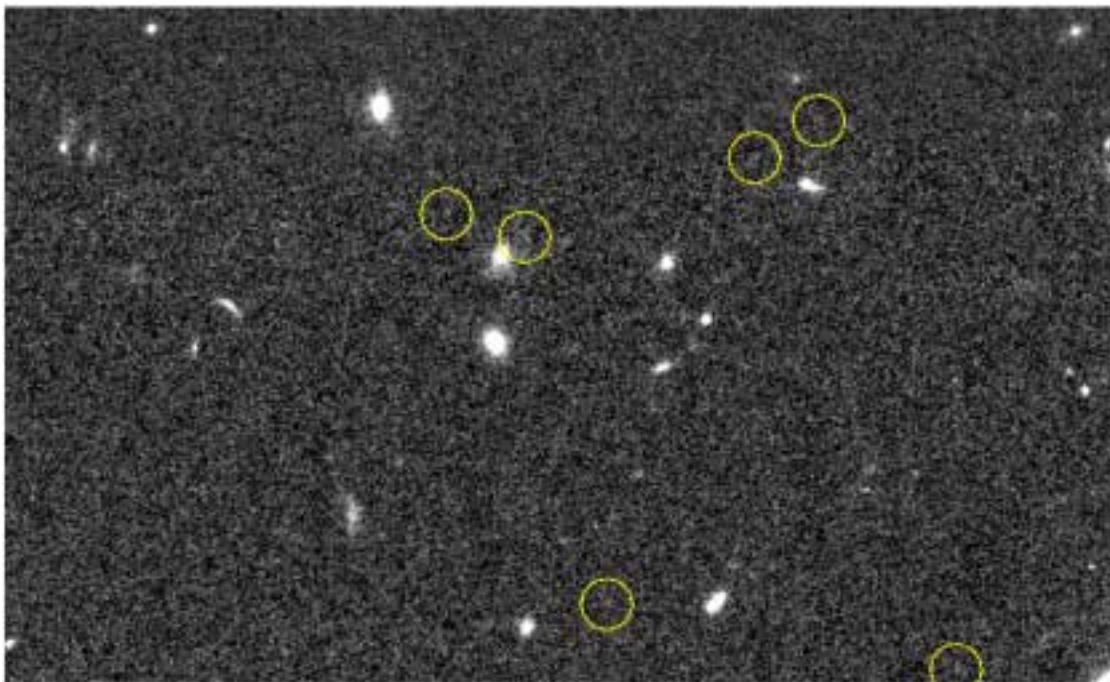


Figure 10: Objects missed in reconstructed image



The brightest one is given as magnitude 26.6 and is the right hand one of the pair of marked sources in upper middle section of Figure 10. It is not clear whether there is a real object at this position or not. So at the extreme lower limit of useful signal-to-noise level, although visually there is no difference between the original and reconstructed image, there are small statistical differences which might occasionally lead to missing a faint object in analysis. A similar test was run retaining four bits of noise data. The compression factor was then close to five but there were still a few sources that behaved in the same way.

The image analyzed above was taken from one epoch of GOODS. Later the same region became available but with five epochs included. Each local area containing a potentially missed source was examined. In none of the cases could a source be seen.

Compression Performance

All the images discussed could be compressed by factors ranging from 4.6 to 9.1. From experiments with several more images I find a typical value of around six using three bits to express the noise level. Using only two bits for the noise level on the WFC 47 Tuc image produced a compression factor of 11.

Image	Original Size in MB	Compressed Size in MB	Compression Factor	Compression Time (sec.)	Decompress Time (sec.)
HRC 47 Tuc	10.5	2.3	4.6	3	2
WFC 47 Tuc	213.5	23.5	9.1	47	29
Light Echo	16.8	2.8	6.0	5	2
Galaxies	67.1	11.4	5.9	17	10

Table 3. Compression factors and times with 3 bits of noise.

The compression rate is about 4 megabytes per second on the above images, with decompression rates being about 50% faster. These rates were measured on a 359 MHz Sparc 60 machine with all files being on a local disk. The rate does not depend on the size of the file. This is to be expected as the method operates on successive portions of the image simply repeating as many times as necessary. The portions used in all these test were single rows of the image. For the same reason I would not expect any great variation in the performance as a function of the computer memory size. The Appendix gives instructions on how to run the program **fcomp** together with information on how the different components of a FITS file are treated.

Summary

The compression method described here appears to be suitable for use in supplying astronomical data. Compression factors of about six are obtained by using three bits to encode the noise level. There is some numerical difference between the original image and its reconstituted form, but except for very faint objects at the detection limit of the image, nowhere did this result in any meaningful difference in the analysis.

Acknowledgements

The program executable **fcomp** and assistance in using it was supplied by Nadezhda Dencheva. I have had useful discussions with Perry Greenfield who supplied me with material and explanations. I particularly wish to thank Swara Ravindranath who provided me with catalogs of the GOODS images generated by the program **SExtractor**.

References

White, R. L. and Greenfield, P. 1999, A Scheme for Compressing Floating-Point Images, Astronomical Data Analysis Software and Systems VIII, ASP Conference Series Vol 172, 1999 p125.

RICE local "28 March 1996",

<http://www.tass-survey.org/richmond/rice/rice.1.html#alg>

GOODS, The Great Observatories Origins Deep Survey,

<http://www.stsci.edu/ftp/science/goods/>

Appendix

Instructions and options for fcomp

This is a fits image compression program, developed by the Science Software Branch at STScI.

Three kinds of compression algorithms are available: Rice, Gzip and Plio

For a detailed description of the format see:

http://heasarc.gsfc.nasa.gov/docs/software/fitsio/compression/compress_image.htm

1

Usage: fcomp [options] infile

Options:

-h[elp] : prints this page
-r[:n] : rice compression for all extensions
-p : pixel list for all extensions
-g : Gzip compression for all extensions
-s[uffix] <suffix_string> : what string to append to main filename body:
e.g. fn_cs.fits if suffix_string is 'cs'
default suffix is 'cmp' for compression,
'dcm' for decompression
-o[utput] <output filename> : works only with one file
-t : sets the size of the tile to the entire image. The default tilesize (with-
out -t) is one row
-t[:n1[xn2[xn3]]] : sets the tile size to be n1xn2xn3
-d : decompresses all extensions

Without any options, all extensions will be compressed with the default options:

defaults:

compression algorithm: RICE

noise bits : 4

Working with extensions:

Pass the '-e' option to compress individual extensions.

Multiple '-e' options are acceptable. The syntax is:

-e:extension:[compression algorithm]:[noise bits]

extension - can be extension name or extension number

compression algorithm (optional) - can be 'r', 'g' or 'p'

noise bits (optional) - used with Rice algorithm

If algorithm or noise bits are not specified, the default ones are used

-e:hst will compress all hst type extensions with default values, which currently are

HST Defaults:

SCI:

compression algorithm: RICE

noise bits : 6

ERR:

compression algorithm: RICE

noise bits : 1

DQ:

compression algorithm: PLIO

noise bits : 4

SAMP:

compression algorithm: GZIP

noise bits : 4

TIME:

compression algorithm: GZIP

noise bits : 4

Use the '-d,' option to decompress individual extensions.

Multiple extension names or numbers are acceptable with ',' as a delimiter. The syntax is:

`-d,extension1[,extension2]` : decompresses a particular extension

Examples:

1. To compress all extensions using RICE with noise bits 3, writing the output to a file, called output.cmp.fits:
`fcomp -r:3 -o output.fits infile`
2. To compress all extensions in all fits files in the current directory, using the default options:
`fcomp *`
3. To compress extension 2 with Rice and noise bits 2 and all extensions with extension name 'ERR' with GZIP, in a file infile.cmp.fits:
`fcomp -e:2:r:2 -e:ERR:g infile.fits`
4. To compress all files with hst type extensions, using the defaults for hst extensions with a tile 100x100:
`fcomp -t:100x100 -e:hst *`
5. To decompress all extensions in all files in the current directory
: `fcomp -d *`
6. To decompress extensions 5 and all extensions with extension name 'SCI'
`fcomp -d,5,SCI infile`