Astronomical Data Analysis Software and Systems XVI **O1.5**
ASP Conference Series, Vol. XXX, 2006
R. Shaw, F. Hill and D. Bell, eds.

Astronomical Tiled Image Compression: How & Why

Rob Seaman

National Optical Astronomy Observatory, Tucson, AZ 85719

William Pence

NASA Goddard Space Flight Center, Greenbelt, MD 20771

Rick White

Space Telescope Science Institute, Baltimore, MD 21218

Mark Dickinson, Frank Valdes, Nelson Zárate

NOAO

Abstract. A tsunami of astronomical data is fast approaching. New survey programs will dwarf any data sets that have come before. Observatories' data storage costs will threaten their budgets. Data transport latency and bandwidth will threaten not just budgets, but available technology and human patience. Under such circumstances, projects will increasingly rely on data compression as a technique necessary to meet requirements of acceptable throughput and affordable data handling.

1. Introduction

The FITS Tile Compression Convention (Pence et. al. 2000) provides a general framework for any compression algorithm that can operate on multidimensional image sections, while preserving access to FITS (Hanisch et. al. 2001) headers and individual FITS HDUs. This convention[1] has been supported by CFITSIO[2] for several years, including Rice, PLIO and gzip compression. The IRAF FITS kernel also supports the PLIO format. Various projects have used these facilities, but as interest in tile compression has grown, limitations have become clear:

- Implementations were not *idempotent* – even a losslessly compressed file would suffer keyword changes upon decompression.
- The original tile compression convention covered only per-HDU issues. For example, compressing a single HDU (SIF) image generated a binary table indistinguishable from an MEF original.

[1] `http://fits.gsfc.nasa.gov/fits_registry.html`

[2] `http://heasarc.gsfc.nasa.gov/fitsio/fitsio.html`

- The only tile compression application was the example imcopy program with limited features. Copying actually increases total size.

Improvements to CFITSIO that address these issues are discussed, including the introduction of the host level *fpack* compression tool. Fpack can compress images in-place. Tile compression can work with other FITS tools to build large multi-image archives to improve transport efficiency. Fpack (and *funpack*) are idempotent, support the FITS Checksum (Seaman 1995), and applications layered on CFITSIO can access output files transparently without decompression. Support for the Hcompress (White & Greenfield 1999) algorithm has been added to CFITSIO, including an improved option for scaling floating point input.

2. Optimize throughput, not just storage

The challenges faced by the National Optical Astronomy Observatory's Data Products Program are representative of other major astronomical institutions. Data flow rates, already large by historical standards, will soon grow by orders of magnitude. NOAO and partner institutions operate telescopes on Kitt Peak in Arizona, and on Cerro Tololo and Cerro Pachon in Chile. Major instruments are being commissioned right now (e.g., NEWFIRM) whose nightly data outputs will match any we currently operate. Two mountaintops will see new instruments (the One Degree Imager at WIYN and the Dark Energy Camera at CTIO) that will increase our data flow ten times in about three years. Another few years beyond, LSST will raise the bar again by a factor of perhaps another thirty.

Table 1 shows the NOAO Science Archive data volumes[3] estimated to accumulate through the end of the next few observing semesters, contingent on the compression algorithm as well as the output datatype used by the NOAO High Performance Pipeline. Varying these can have a large effect on storage costs, for example, the additional cost for specifying 32-bit floating point output and the gzip[4] algorithm is $2.86[5] per input Mosaic-camera image versus Rice[4] compressed short (16-bit) integers. Floating point output suffers from doubled BITPIX, but is also less compressible than integer output. The NOAO test data realized 57% integer gzip compression, but only a 74% benefit for floating point.

Static storage costs are only one consideration. Raw NOAO data streams (see Figure 1 in Smith et. al. 2006) flow down from the three mountaintops to data centers in Tucson, Arizona and La Serena, Chile. Data are mirrored across the equator between the two data centers. Data from key instruments feed into pipelines that may inflate sets of large multi-array input files by a factor of several. All data are also forwarded to the NCSA in Illinois for permanent MSS storage. Each data product that is ingested thus requires multiple replications across a far-flung network. Any archive must also plan for success and one hopes that each ingested data product will be retrieved many times over by

---

[3] These figures do not include data from instruments yet to be commissioned in the near-to-midterm such as NEWFIRM, WHIRC, QUOTA, ODI and DEC, nor data from SOAR.

[4] *gzip* is the standard Unix gzip command, *Rice* is FITS tile compression with default settings.

[5] Accounts for the mirror archive in Chile and the wide variety of pipeline data products.

Table 1. Cumulative NOAO Science Archive holdings in TBytes, both raw and pipeline processed data, at the end of indicated semesters

comp. algor.	pipeline BITPIX	now 1Aug06	2006B 1Feb07	2007A 1Aug07	2007B 1Feb08	2008A 1Aug08	2008B 1Feb09
none	-32	30.6	40.1	47.8	57.3	65.0	74.5
gzip	-32	20.3	26.6	31.6	37.9	42.9	49.2
none	16	22.4	29.2	34.8	41.6	47.2	54.0
gzip	16	13.8	18.0	21.4	25.6	29.0	33.2
Rice	16	11.5	15.0	17.8	21.3	24.1	27.6

users. In short, the value of data compression is multiplied many times over when budgeting for network bandwidth.

For interactive applications, such as data access through an NVO portal, minimizing network latency may be even more important. Users want instant gratification – moreover, their real time science may demand it. Each network copy operation is reflected in an increased data delivery delay. Each unnecessarily uncompressed byte adversely affects customer satisfaction.

The end-to-end goals thus become to support cradle-to-grave compression (from "dome to home"), to retain easy FITS header access (no external caching needed), to optimize pixel access (for inexpensive cutouts and mosaicing), and to support read-write speeds as fast (or faster) than when uncompressed.

3. Compress once, decompress never

The alternative to FITS tile compression is not some host compression algorithm like gzip, rather the alternative is no compression at all. No single host level algorithm can generate enough interest to gain support across all classes of astronomical applications. By building compression into FITS, FITS compliant applications will evolve to suit the growing requirements of bold new projects.

No one compression type is best, but consider the familiar gzip (or bzip2) tools. Support for these tools is ubiquitous, but so are their shortcomings. The selection of any compression algorithm keys on three main points: the compression factor achieved, the speed of compression, and the speed of decompression. As shown in table 1, FITS tile compression using the Rice algorithm beats gzip file sizes by 17% for a large sample of short integer NOAO data products. But Rice also beats gzip for speed of both compression (quite significantly) and of decompression. For the same data sample, fpack using Rice compressed in only 39% of the time required by the Uniz gzip command. Gzip decompresses much faster, a factor of about 30%, than it compresses, but Rice still edges it out at 28%, normalized to gzip compress. Data handling thoughput thus benefits from both smaller size and quicker (de)compression.

Logistical concerns may trump even these impressive benefits. When using a host compression utility, the contents of each file are opaque. Headers and other metadata, pixels and tabular data are all unreadable. For most purposes, a compressed file must be decompressed before use. Software provides only limited access to on-the-fly decompression for gzipped FITS files – e.g., libraries cannot seek to extension headers without expanding intervening bytes. Paradoxically, network transport is most efficient for large files, such that compressed files must be assembled as collections, or contrarily that file collections themselves be compressed (hence the familiar ".tar.gz"). Complexity and overhead result.

By contrast, the FITS tile compression convention provides a standard framework that can support any algorithm that operates on multidimensional image tiles (sections). With the addition of Hcompress, this now includes very high compression factors through lossy techniques. Most importantly, a tile compressed image (while represented as a binary table) looks exactly like an uncompressed image to CFITSIO for both reading and writing – in fact, a new image can be created already compressed. In addition:

- FITS headers remain readable and writable
- access is preserved to individual FITS HDUs
- the files are still FITS and any FITS operation is legal

In short, while *funpack* may be used as a host level tool similar to *gunzip*, better yet, **data need never be decompressed** for CFITSIO applications. For example, it is trivial to assemble mixed collections of compressed and uncompressed HDUs into arbitrarily large FITS MEF files.

4. New features supported by CFITSIO and *fpack*

- compression is now idempotent (decompression restores verbatim original)
- applications layered on CFITSIO access compressed files transparently
- both lossy and lossless Hcompress algorithms are supported
- the input scaling of floating point data can be adjusted
- *f(un)pack* (de)compresses files in-place or via a pipe
- memory consumption has been optimized for per-HDU operations
- *fpack* and *funpack* update the FITS Checksum
- works with multi-file, multi-extension archives for transport efficiency

References

Hanisch, R. et. al. 2001, A&A, 376, 359

Pence, W. et. al. 2000, in ASP Conf. Ser., Vol. 216, ADASS IX, ed. N. Manset, C. Veillet, & D. Crabtree (San Francisco: ASP), 551

Seaman, R. 1995, in ASP Conf. Ser., Vol. 77, ADASS IV, ed. R. A. Shaw, H. E. Payne, & J. J. E. Hayes (San Francisco: ASP), 247

Smith, R. C. 2006, this volume, [O6b.1]

White, R. & Greenfield, P. 1999, in ASP Conf. Ser., Vol. 172, ADASS VIII, ed. D. M. Mehringer, R. L. Plante, & D. A. Roberts (San Francisco: ASP), 125