Volume Title
*ASP Conference Series, Vol. **Volume Number***
Author
*© **Copyright Year** Astronomical Society of the Pacific*

A New Compression Method for FITS Tables

William Pence[1], Rob Seaman[2], and Richard L. White[3]

[1]*NASA Goddard Space Flight Center, Greenbelt, MD 20771*

[2]*National Optical Astronomy Observatories, Tucson, AZ 85719*

[3]*Space Telescope Science Institute, Baltimore, MD 21218*

Abstract.
As the size and number of FITS binary tables generated by astronomical observatories increases, so does the need for a more efficient compression method to reduce the amount disk space and network bandwidth required to archive and download the data tables. We have developed a new compression method for FITS binary tables that is modeled after the FITS tiled-image compression convention that has been in use for the past decade. Tests of this new method on a sample of FITS binary tables from a variety of current missions show that on average this new compression technique saves about 50% more disk space than when simply compressing the whole FITS file with gzip. Other advantages of this method are (1) the compressed FITS table is itself a valid FITS table, (2) the FITS headers remain uncompressed, thus allowing rapid read and write access to the keyword values, and (3) in the common case where the FITS file contains multiple tables, each table is compressed separately and may be accessed without having to uncompress the whole file.

1. Overview

Modern astronomical observatories continue to produce greater volumes of data in FITS binary table format, for example in the form of photon event lists and in mega- and giga-sized object catalogs. Currently, the only widely available method for reducing the storage size of these data tables is to compress the entire FITS file using an external file compression utility such as gzip or bzip2. We have developed a new compression method specifically designed for FITS binary tables that generally produces significantly higher compression. In addition, this new method has the advantage that the FITS headers remain uncompressed, which allows rapid read and write access to the keywords. Also, in cases where the FITS file contains multiple binary tables, each table is compressed separately and may be directly accessed without having to first uncompress the entire FITS file.

This new binary table compression method is modeled after the FITS tiled-image compression convention (see the FITS Support Office site at http://fits.gsfc.nasa.gov/registry/tilecompression.html) that has been in use for about a decade. This present paper provides a high-level description of this compression method and some preliminary performance results on a sample of FITS binary tables. Further technical details about this proposed FITS table compression method can be found in a draft specification document also available on the FITS Support Office web site.

2. Description of the Compression Method

The algorithm for compressing a FITS binary table is described in the following subsections. Only the tabular data values are compressed, and the header keywords remain uncompressed for rapid access. Almost all the keywords in the uncompressed table are copied verbatim into the header of the compressed table with only a few exceptions (such as the NAXISn and TFORMn keywords which necessarily define the structure of the compressed table itself).

2.1. Divide the Table into Tiles (Optional)

In order to limit the amount of data that must be manipulated at any one time, large FITS tables may be optionally divided into blocks, or tiles, where each tile contains an equal number of rows, except for the last tile which may contain fewer rows. Each tile is compressed in sequence, and is stored in a row in the output compressed table which is itself a valid FITS binary table. There is no fixed upper limit on the tile size, but it is recommended that FITS tables larger than about 10 MB in size be divided into multiple tiles so as to not impose too large of a memory resource burden on software that must uncompress the table.

2.2. Transpose the Rows and Columns

The data cells in a FITS binary table are stored in row by row order, so that the values in the first row of every column are given in order, followed by the values for the second row, and so on. It can be difficult to efficiently compress this native stream of data values, however, because binary tables can contain very heterogeneous types of data in different columns. For this reason, almost all FITS tables can be better compressed by first internally transposing the rows and columns in the table so that all the values for the first column occur first, followed by all the values for the second column, and so on. Then the data values within each column can be compressed separately.

2.3. Compress Each Column

After transposing the rows and columns in the table, each column of data is compressed with a suitable compression algorithm. Our prototype table compression utility program uses the gzip compression algorithm by default for all columns, but in principle each column could be compressed using a different algorithm that is optimized for that particular type of data (e.g., using the Rice algorithm for integer columns).

Our prototype compression code also preprocesses numeric column values (integer and floating-point) using a “byte shuffling” technique which rearranges the bytes so that the most significant byte of every column element appears first, followed by the next most significant byte of every element, and so on. Since the most significant byte(s) of each column value often contains nearly the same bit pattern, these bytes tend to compress very well when reordered in this way. Then when the table is uncompressed, the inverse shuffling is applied to restore the original byte order of all the column values.

In most cases a lossless compression algorithm will be required in order to exactly preserve the values in each column. In specialized cases, however, a lossy compression algorithm could be used to achieve higher compression if the values in a particular column only need to be approximately represented.

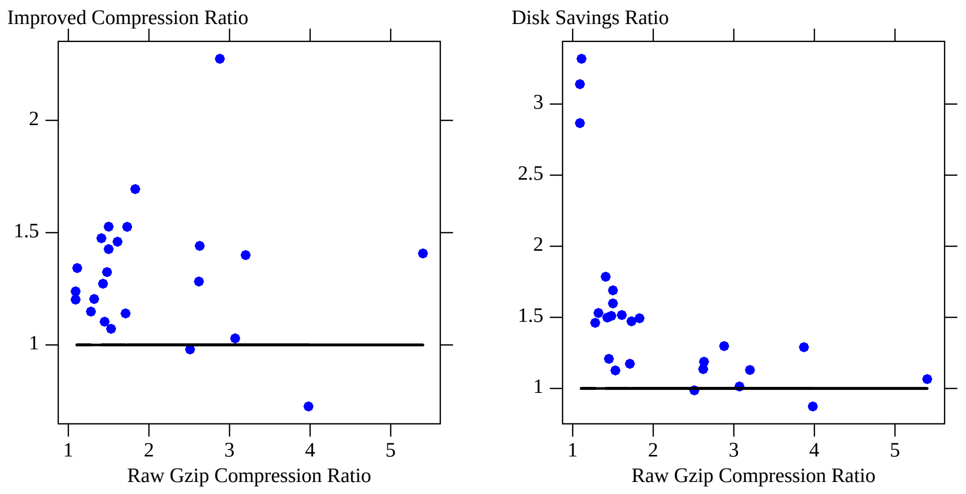

Figure 1. (Left): The points show the factor by which the compression ratio is improved when using the new compression method (y-axis), as compared to simply compressing the whole file with gzip (x-axis) for the 24 tables in our sample. (Right): The ratio of how much more disk space is saved when using the new compression method, as compared with just using gzip.

2.4. Store the Compressed Bytes

The compressed stream of bytes for each column is stored in the corresponding column in the output table, which is itself a valid FITS binary table with variable-length array columns. Each row in the output compressed table corresponds to one tile of rows in the uncompressed table. If the input table is compressed as a single tile, then the output table will only contain one row.

3. Example Results

We have implemented a prototype version of this compression method and applied it to a sample of 24 different types of FITS binary tables produced by 9 different missions (Chandra, Dark Energy Camera, Fermi, Herschel, PanSTARRS, ROSAT, SDSS, Swift, and XMM). For these tests we used the gzip algorithm to compress every column, after applying the previously described byte shuffling algorithm to the numeric column values. We compared the compression ratio of the tables when using this new method to that of the standard technique of simply compressing the whole FITS table with the gzip utility program. All the sample FITS tables are very large, so the fact the FITS headers remain uncompressed using our new method has no significant effect on the overall table compression ratio.

Figure 1a shows that in 22 out of the 24 cases, the technique of transposing the rows and columns in the table and shuffling the bytes in the numeric columns before compressing the column with gzip, produces better compression than simply using gzip to compress the raw FITS binary table. On average, our new method produced 1.3 times more compression then just using gzip alone. In only 1 case did the table compress significantly less well after shuffling the bytes in the numeric values. A more sophisticated approach would be to only apply the byte shuffling technique in cases where it actually improves the subsequent compression ratio. The CPU times needed to compress or uncompress these tables using the new method is not much greater (less than a factor of 2) than when simply using gzip, because the additional time needed to transpose the table and shuffle the bytes is relatively small compared to the time needed to actually compress the bytes with the gzip algorithm.

Of more direct practical interest, Figure 1b shows the ratio of how much more disk space is conserved by using our new method compared to just gzipping the whole file. On average, the new method saves about 50% more disk space than when just using gzip. The biggest disk space gains are for the files that compress by less than a factor of 2.

4. Summary

These tests demonstrate that this new compression method can significantly reduce the network bandwidth and disk space needed to transfer and archive the large volumes of FITS binary tables produced by current and future astronomical missions. In addition, the fact that the header keywords remain uncompressed, and that individual tables within a multi-table FITS file can be directly accessed without uncompressing the whole file should make this an attractive on-line data analysis format. We plan to conduct additional tests on a more comprehensive set of astronomical FITS tables to further optimize the compression strategy for different types of tables.