MNRAS **000**, 1–12 (2020) Preprint 11 February 2020 Compiled using MNRAS LATEX style file v3.0

SnowPac: A Multi-scale Cubic B-spline Wavelet Compressor for Astronomical Images

Jesus Pulido,[1]★ Caixia Zheng,[2]† Paul Thorman[3] and Bernd Hamann[1]
[1]*Department of Computer Science, University of California, One Shields Avenue, Davis, CA 95616, USA*
[2]*College of Information Sciences and Technology, Northeast Normal University, 2555 Jingyue Street,Changchun 130117, China*
[3]*Departments of Physics and Astronomy, Haverford College, 370 Lancaster Avenue, Haverford, PA 19041, USA*

Accepted 2020 February 7. Received 2020 February 2; in original form 2019 January 9

ABSTRACT
As more advanced and complex survey telescopes are developed, the size and scale of data being captured grows at increasing rates. Across various domains, data compression through wavelets has enabled the reduction of data size and increase in computation efficiency. In this paper, we provide qualitative and quantitative tests of a new wavelet-based image compression method compared against the current standard for astronomical images. The analysis is improved by making use of state-of-the-art object detection systems to accurately measure the impact of the compression. We find that a combination of lossy wavelet-based methods, efficient quantization, and lossless dictionary compressors can preserve up to 98% of astronomical objects at a 10:1 compression ratio. This significant reduction in file size also preserves astronomical-object properties better than existing methods. These methods help further reduce future work-loads for image-heavy processing pipelines.

Key words: astrometry – methods: data analysis – techniques: image processing

1 INTRODUCTION

The Large Synoptic Survey Telescope (LSST) is a wide-field survey telescope that is currently being constructed and is projected to produce hundreds of gigabytes of data per night. Transferring and processing raw data daily becomes challenging as network and I/O systems become bottlenecks for this large data transfer problem. In the past, lossless data compressors provided sufficient data reduction but as datasets become larger and more complex, lossy methods have become a necessity. Both transferring data between on-site systems and sharing data for collaborative research imposes significant bottlenecks making smart but lossy data reduction schemes attractive.

The Flexible Image Transport System data format (FITS) is the standard astronomical data container used in large by the astronomy community. The subject of astronomical image data compression is not entirely new.

There are a range of tools for generating FITS files that employ different compression techniques. CFITSIO is a tool developed by Pence (1999) that provides subroutines for reading and writing FITS data files. A more recent co-development has been integrated called fpack and has been proposed for FITS file compression (Pence et al. 2011). This tool provides several general purpose compression algorithms: Gzip, Rice, HCompress, and PLIO. The most popular and simplest method used for general lossless compression is Gzip (GNU 1997). Although lossless, Gzip provides low compression ratios for large, dynamic datasets. Generally, the Rice compression algorithm (Rice et al. 1993) provides better lossless compression ratios and is faster than Gzip. HCompress (White et al. 1992) can be either a lossy or lossless compressor based on the H-transform (Fritze et al. 1977). HCompress uses a generalized 2D Haar, quantization, and quad-tree coding to achieve its lossy compression. This method generally compresses slightly better than Rice at the cost of more compute time.

Previous work by Pence (2009) conducted a feasibility study for the use of a more efficient compression algorithm named BZip2 in astronomical images. That work concluded that BZip2 was an order of magnitude slower compared to Rice and required significantly larger block sizes to achieve same compression efficiency. Unlike other compressors, JPEG2000 is keyed as a true image compression standard that intends to take advantage of multidimensional data. Similar applications by Peters & Kitaeff (2014), Kitaeff et al. (2015), and Vohl et al. (2015), analyzed the impact of JPEG2000 lossy compression on radio astronomy imagery. Since JPEG2000 is a non-standard method for astronomical images, several challenges may appear such as the inability to process floating-point images, requiring the conversion to

★ E-mail: jpulido@ucdavis.edu
† E-mail: zhengcx789@nenu.edu.cn

integers prior to processing. This simplification of the data may lead to issues when handling images with very high dynamic ranges in intensities. Methods such as those in Vohl et al. (2017) overcome this by adding 32-bit support, but do not go in depth on the effects of achieving 'extreme' 35:1 compression. Compared to traditional observational sky survey data, radio astronomy imagery is significantly noisier and lacks the structures needed for general purpose compression algorithms to excel.

Other methods have suggested the deviation from the FITS file format to a more compression-friendly container such as HDF5 in Price et al. (2014) and Masui et al. (2015). The advantage of switching to HDF5 is its availability of internal data filters. One such filter explored in both works is BitShuffle, a lossless compressor that shifts binary data for more efficient encoding. As with other lossless compressors, BitShuffle is hardly able to achieve a compression ratio beyond 2:1, unable to achieve as much compression as other lossy methods. Finally, singular value decomposition methods (SVD) such as those in Kolev et al. (2012) and Morii et al. (2017) are used to compress astronomical movie data in a lossy representation.

When lossy compression is used, the goal is to preserve as many astronomical objects as possible without significantly altering their science-critical characteristics, such as the total flux, the spatial extent and profile of the flux, and the orientation of the source profile on the sky. In a typical pipeline, object detection and extraction is performed by widely used tools such as Source Extractor by Bertin, E. & Arnouts, S. (1996) or the techniques of Zheng et al. (2015), which attempt to extract and characterize these astronomical objects.

The work presented in this paper introduces a new wavelet-based lossy image compression method for astronomical images. This new method is able to achieve extreme levels of compression while preserving features of interest (astronomical objects). To accurately measure the impact of compression, state-of-the-art detection systems are used to accurately measure differences, both qualitatively and quantitatively.

This paper is structured as follows: Section 2 gives an overview on the construction of the method and usage to achieve these results. Section 3 evaluates the performance, cost, and accuracy of our method by comparing the obtained results with those of competing methods. Section 4 provides conclusions and points out possible directions for future work.

2 METHOD

2.1 Implementation

A combination of several techniques can be used in a pipeline; see Figure 1, where the data compression pipeline is depicted with its main components marked in the center. Once a FITS image is read, the largest data reduction occurs in the initial wavelet decomposition and wavelet coefficient floating-point to integer quantization. Using the native hierarchical structure generated by a wavelet decomposition, the highest-magnitude wavelet coefficients are preserved with high precision and the lowest-magnitude at a lower precision via quantization. Once completed, a lossless encoding step is performed where, optionally, further compression can be achieved, at the cost of compute time. Like compression, decompression only requires us to decode the stored coefficients and perform a wavelet reconstruction, returning the coefficients to real-space.

We present a data reduction pipeline with flexible components, trading off lower file sizes for higher computation time. As a result of the compression, a significantly larger subset of astronomical imagery can be stored, processed, and transferred. The complete method named *SnowPac* (SpliNe Wavelet Packing And Compression) (Pulido 2019) is made available in both C++ and Matlab variants. The end of a typical processing pipeline may contain additional astronomical object detection components that quantify the number of stars and galaxies in a certain region of the sky. The usage of wavelet methods in our pipeline leads to advantageous properties such as data streaming, selective decompression of data subsets, and denoising of astronomical images.

2.2 Wavelet Compression

The wavelet transform is a generalization of the Fourier transform, using bases that represent both location and spatial frequency (Daubechies 1992). Typical wavelet signals contain several vanishing moments that allow for a sparse but accurate representation of an input dataset, with only a small number of coefficients. A signal is decomposed through multiple steps involving "folds" at the largest scale until data is reduced to the smallest scale. A 2-D example for an astronomical image is shown in Figure 2. Here, a fold is performed twice for each dimension until a set of low- and high-pass coefficients is obtained. The upper-left quadrant demonstrates the capability of wavelets to preserve coherent features (astronomical objects) and, at the same time, to filter out small-scale signal behavior usually correlated to Gaussian noise. Additionally, this behavior creates a multi-scale structure where each fold represents features at a certain resolution. This capability allows a compressor to reduce data to extreme levels without losing important objects, even at medium to high levels of dynamic range. Additionally, a 2-D transform can be expanded to 3-D data arising in multiple applications, such as stacked imagery, supporting the signal acquisition and preservation of many images of the same sky region.

Unlike the classical and simple Haar wavelet, biorthogonal B-spline wavelets are an extension with similar usage and implementation, providing the capability to capture more complex behavior via their basis functions (Cohen et al. 1992).

In a recent study by Pulido et al. (2016), several multiresolution representation methods, including higher-order B-spline wavelets, were tested for their ability to capture a broad range of quantities pertaining to turbulent data representation using a reduced set of coefficients. The higher-order B-spline families consistently scored at or near the top based on the metrics considered. One of the most important metrics was the preservation of coherent structures using the least number of coefficients through hard thresholding. Here, the fourth family of B-spline wavelets (cubic) is considered as the method of choice for compression and analysis. A cubic B-spline wavelet transform can capture and preserve

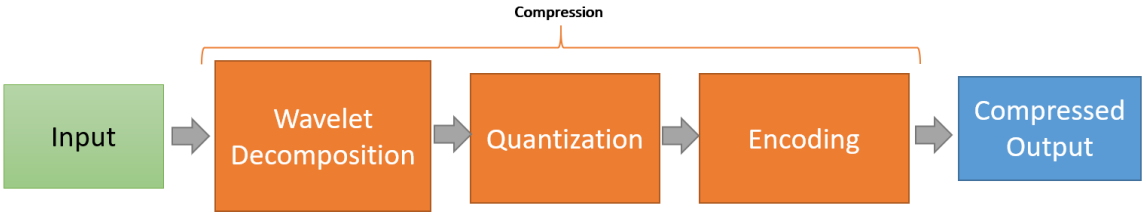

Figure 1. The image compression pipeline has an image as input, applies cubic B-spline wavelet decomposition, and performs floating-point to integer quantization for an initial lossy step. After, the encoding step efficiently stores lossy coefficients in a lossless format.

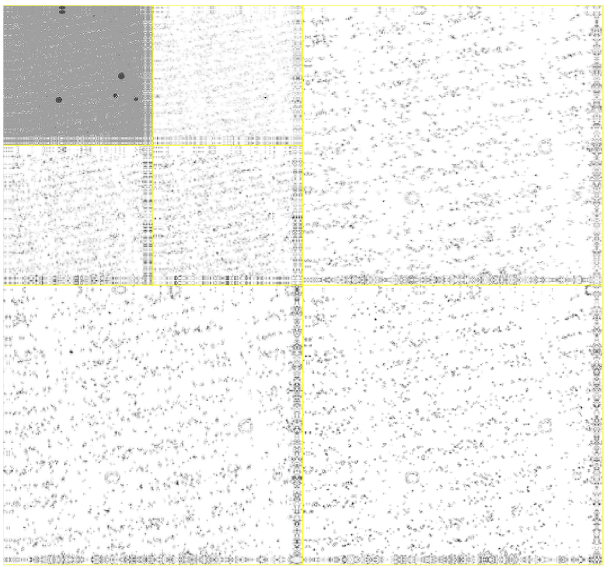

Figure 2. 2D data decomposition. Decomposition of an astronomical image using two levels produces a series of low- and high-pass coefficients using cubic B-spline wavelets. Astronomical objects are preserved, while small-scale noisy artifacts are filtered out.

directionality well. This characteristic is particularly important in the effort to preserve the structures of astronomical objects with high accuracy. We employ the discrete version as described by Shensa (1992) in our processing pipeline.

2.3 Hard Thresholding and Quantization

Hard thresholding selects only distinct coefficients up to a certain cut-off threshold, while setting to zero all other coefficients, after a wavelet decomposition has been performed. The primary criterion for selecting a cut-off threshold is a coefficient's (absolute) magnitude. Coefficient magnitude can be understood as signal energy for a frequency (or basis function). Thresholding is done by sorting the coefficients by magnitude and selecting the largest ones, either via a cut-off percentage or a threshold value.

This lossy compression approach is not sufficient for achieving a large-enough reduction for astronomical images. Controlling the precision of a quantized data representation of coefficients is preferred.

Figure 3 shows a comparison between hard thresholding for a fixed percentage of coefficients (10% relative to original size) and a floating-point-to-integer quantization for a compression ratio near 10:1 and the effect on object detection. Both reduction techniques produce similar results in terms of detected objects, but there are differences. Most significantly, an object's shape is better preserved when using all wavelet coefficients at a reduced precision rather than using a subset of large-magnitude coefficients via hard-thresholding.

Original Value: 116618.821101			
Weight	Shifted	Q Integer	Bits
0.01	1166.188211	1166	12
0.1	11661.88211	11661	15
1	116618.8211	116618	18
10	1166188.211	1166188	22
100	11661882.11	11661882	25

Table 1. Floating-point-to-integer quantization. Using a weight as input, a quantization operation shifts the decimal during conversion to control the amount of bits needed for storage.

Figure 4 shows the coefficients sorted by magnitude. For a typical sky survey dataset, many low-magnitude coefficients exist that can be reduced in precision via quantization, with little impact as long as high-magnitude coefficients are preserved with higher-precision. This principle is used to achieve lossy compression.

A wavelet transform produces a floating-point dataset. Typically, FITS images are integer data. First, we convert them to floating-point representation. For efficient data reduction we apply floating-point-to-integer quantization across all wavelet coefficients (see Section 2.3). By performing re-quantization, we efficiently encode integer coefficients.

We use a highly efficient floating-point-to-integer quantization method that shifts the decimal of the floating-point representation to the right, to preserve the integer portion and truncate the fractional half. This method is available in HDF5's (Folk et al. 2011) "scale_offset" filter and ensures minimal additional loss in precision when combined with wavelets, as most of a coefficient's entropy is captured in its integer component. For example, a coefficient value of 3027.567812346 is converted to the integer 3027567, using a three-decimal scale off-set shift of 1000. Table 1 lists the number of bits used to represent a floating-point value as an integer.

This technique allows us to control the precision of coef-

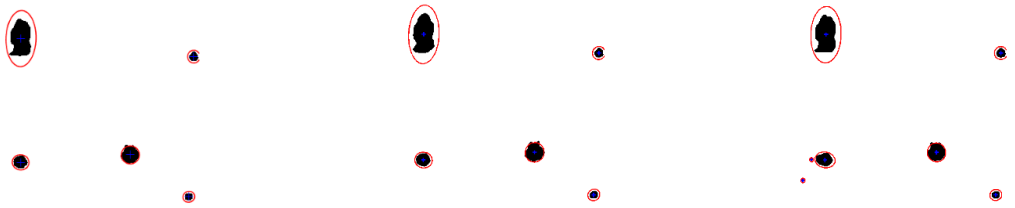

Figure 3. Coefficient quantization vs. thresholding. The shown objects result from performing lossy coefficient operations, illustrating their preservation properties. Quantization of all coefficients using a 10:1 ratio (left), hard thresholding using 10% coefficients (center), and original data (right) show that pronounced stars and galaxies indicated by red ellipses are preserved after a binary image operation during object detection.

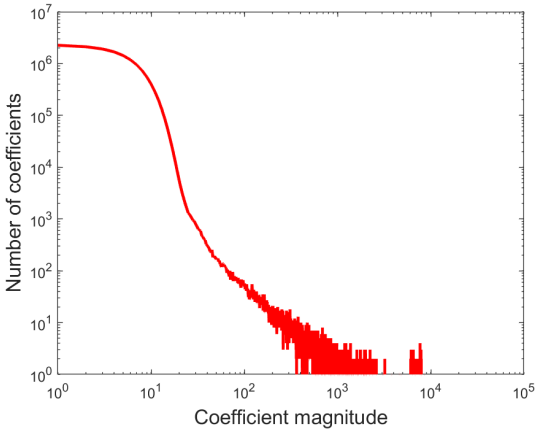

Figure 4. Coefficient magnitude sorting. This plot shows the magnitude-sorted cubic B-spline wavelet coefficients for an LSST dataset. Many low-magnitude redundant coefficients exist, making posisble a lower-precision representation using quantization without significant data information loss.

ficients. By using the natural wavelet hierarchy, coefficients at the lowest bands have the largest magnitudes and importance. Therefore, one should preserve them at higher precision compared to those at the highest bands, having lowest magnitudes. Example weights for an extraction of eight levels are [100, 10, 10, 10, 10, 10, 1, 1], where more bits are used at the lowest and less bits at the highest levels.

2.4 Encoding

The simplest and most common encoding method available is Run-Length Coding (RLE), which efficiently packs wavelet coefficients after quantization. RLE counts the number of repeated byte spaces in a dataset and compacts the representation with a single value.

As a more valuable option, we use off-the-shelf, dictionary-based lossless compression methods to expand the overall capabilities of our method. The Lz4 method (Collet 2011) is characterized by its extremely high speed, coming at the expense of low compression ratios; the BZip2 method (Steward 1996) leads to high compression ratios, coming at the expense of low speed. The Gzip technique, traditionally applied to sky survey datasets, exhibits performance that lies somewhere in the middle, producing average compression ratios and computation times. When combined with the floating-point-to-integer quantization method used in the previous section, applying lossless compression on the lossy represented coefficients demonstrates to be an efficient way to pack coefficients for storage.

2.5 Decompression

Decompression of astronomical images using the pipeline shown in Figure 5 takes much less effort that compression. Decompression merely decodes the coefficients for reconstruction and reapplies inverse weights for requantization.

2.6 Object Detection

Analysis on astronomical imagery is typically performed after data acquisition, including object detection of stars and galaxies. By using these methods, we can quantify the impact on compression and decide which method best preserves additional properties in images.

In Zheng et al. (2015), an improved method for detecting objects of interest (galaxies and stars) in astronomical images was recently presented. This new method combined various global and local subroutines to improve detection over existing methods. In this work, after a global detection scheme is applied, refinement is done by dividing the entire image into several irregularly sized sub-regions using the watershed segmentation method. A more refined detection procedure is performed in each sub-region by applying adaptive noise reduction and a layered strategy to detect bright objects and faint objects, respectively. Finally, a multi-threshold technique is used to separate objects that initially blended together. On both simulated and observational data, this method detected more real objects than Source-Extractor at comparable object counts. The method also had an increased chance of successfully detecting very faint objects, up to 2 mag fainter than Source-Extractor on similar data. Due to the improved detection properties of this method and its measurable ability to extract faint objects, it is best suitable for testing compression.

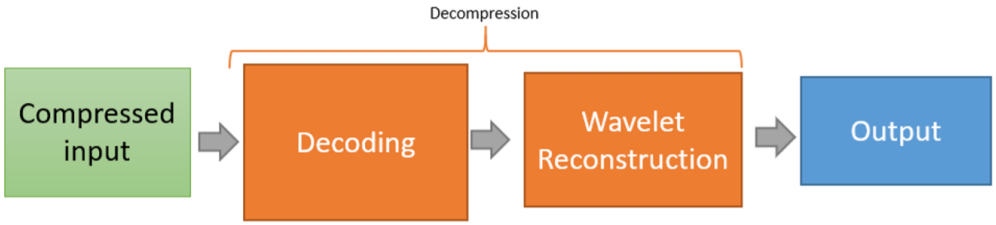

Figure 5. Decompression pipeline. The decompression pipeline requires much less effort than the compression pipeline. Decompression can be performed efficiently and, in some cases, can omit reconstruction altogether for fast data analysis.

3 TESTS AND RESULTS

3.1 Datasets

The following datasets are used to evaluate the accuracy of these methods.

3.1.1 DLS Dataset

The DLS dataset (Wittman et al. 2002) used for testing is taken from a deep *BVRz*' imaging survey that covers 20 square degrees to a depth of approximately 28th magnitude (AB) in *BVR* and 24.5 in z(AB). Our subsample includes four FITS files of size 4096 *x* 4096. The survey was taken by the National Optical Astronomy Observatory's (NOAO) Blanco and Mayall four-meter telescopes. The images correspond to four waveband filters of the same sky area: R, V, B, and z.

Due to the observational nature, there is no ground-truth catalog. To generate one, we compare the objects detected in different bands to gather a coinciding subset of detected objects which are likely to be real. Since the *R*-band is used as a detection band by the DLS survey due to its excellent depth and seeing, detection results of other bands are often verified against it. To compare all types of compression methods, we focus on the *R*-band FITS file [1] due to its prominence in sky survey analysis. In addition, for several levels of lossy compression we detect objects and their properties to compare to the original.

3.1.2 LSST Dataset

The LSST dataset[2] is a simulated set of images that use target parameters for the future LSST project. Using GalSim, the lens properties and other characteristics of the telescope can be simulated to produce expected outputs for the upcoming observational datasets. Our sample set of FITS files are of size 4000 *x* 4000 containing several bands. Unlike the DLS dataset, which represents a series of stacked images, this simulated data has noise added to replicate that of the future telescope. Each pixel in an image represents about 0.2" of real space, so we can expect a single fits file to represent 800" x 800" or 0.05 square degree depth covered by these images.

The LSST data contains the simulated raw output of a single survey image therefore it is difficult to run object detection for validation on it. Without a form of pre-processing, object detection methods will introduce a significant amount of false-positives without stacking. The object detection method used for this analysis is intended to be used on stacked-sets of images. As a pre-processing step, we've emulated the effects of image stacking on one of the regions for the LSST data, which in-turn removes the most obvious pixal-sized artifacts. Thereafter, we can run compression and object detection schemes on this data.

Compression Type	Size (MB)	CRatio	Time (s)
none	64	1.0	–
lz4	17.92	3.57	11.87
gzip	15.04	4.26	18.89
bzip2	9.54	6.71	14.42
SnowPac	6.55	9.77	10.32

Table 2. Comparison between compression methods. Traditional lossless compression methods result in larger file sizes, while our lossy method achieves a high compression ratio while preserving 98% correct objects.

3.2 Compression Types

In Section 2.4, we discussed the usage of various alternative lossless compression schemes to efficiently encode our coefficients. One might ask, why not use these directly on the original data? Table 2 explores that comparison between these traditional compression techniques alongside the wavelet encoding pipeline we've introduced in this paper.

Compared to the alternative lossless methods, we can use our lossy method combined with the high efficiency of Bzip2 encoding to achieve compression ratios as high as 9.77:1 preserving over 98% correct objects, compared to standalone applications of lz4 (3.57:1), Gzip (4.26:1), and BZip2 (6.71:1). Additionally, because the quantization of our coefficients reduces the complexity of the representation of the data through bit reduction, BZip2 operates much faster compared to the original data. As a result, our method overall, i.e. computation of the B-spline wavelet transform, quantization, and encoding, operates faster when paired with BZip2 (10.32s) compared to lossless BZip2 standalone on the original data (14.42s). When comparing a Matlab implementation of SnowPac against the lossy methods implemented

[1] http://dls.physics.ucdavis.edu/imdownload.html
[2] https://www.lsst.org/scientists/simulations/phosim

in C inside of FPack, we found that both HCompress and Rice to be faster in compute time. A faster, more optimized C++ version of SnowPac is planned to be released in the future to achieve faster performance compared to the current Matlab version.

An additional advantage our method is the ability to achieve further compression by reducing the precision of coefficients further, and is explored in the follow-up sections. When the wavelet coefficients are quantized to integers, the complexity of the data is reduced therefore running an off-the-shelf compressor such as BZip2 or LZ4 will reduce the overall compression time compared to running them on the original dataset. The outcome is that our method has a lower total compute time and a higher compression ratio at the cost of losing numerical precision.

The choice of the quantization precision (weights) for the smallest coefficients has the most impact on compression ratios on datasets, since that is the step of our method that poses the most risk of losing data fidelity. These changes are evaluated in the next sections.

Besides lossless methods, we also compare against lossy methods available in fpack (Pence et al. 2011), which include HCompress and Rice. Both methods were tested using recommended parameters from the fpack user manual, which include Rice quantization levels of 8, 4, 2, 1, 0.5 and HCompress scales of 0.5, 1, 1.5, 2, 2.5, 3, 4 and a texture size of 256 *x* 256. The lossy compressor PLIO is available in fpack, but was unable to process our both datasets reporting a '*data out of range for PLIO compression (0 - 2**24)*' error. In addition, the JPEG2000 implementation available in Matlab was unable to process floating point FITS imagery. Due to these issues, both of these compressors were ommitted.

3.3 Evaluation – Dataset 1

We evaluate the impact of our compression scheme by controlling the quantization amounts for wavelet coefficients on the DLS dataset. We then compare several quantities related to the detection of objects in the imagery, such as the number of objects detected and preserved compared to the original, uncompressed data set.

Figure 6 compares the ratio of correctly detected objects per compression ratio relative to the original dataset. The accuracy ratio is a function of objects that are verified against the ground truth, over the total number of objects in the ground truth. The compression ratio is a function of the original file-size over the compressed file-size. When comparing our method against HCompress and Rice, we are able to achieving a higher compression ratio while preserving a significant amount of correct objects. Additionally, our method is able to achieve extreme levels of compression (25:1 ratio) and stay above an accuracy ratio of 0.925. We observe a stair-case effect with HCompress, where accuracy is greatly affected for decimal-valued scales such as 0.5, 1.5, and 2.5.

To compare all methods as equitably as possible, we've selected the compressed datasets closest to a 10:1 compression ratio as seen in Tab. 3. From this data, the original ground truth contained 1433 objects. The general behavior across all lossy compression methods shows that the total number of detected objects increases as more noise is introduced through data precision loss. As expected, the number of correctly detected objects drops and increases the likelyhood of false positives (FP). Our method shows that we are able to achieve the highest compression ratio, reduce the number of false positives in detected objects, and achieve near the highest accuracy (True Positives) for correct objects while compressing an additional 2.00 ratio over HCompress.

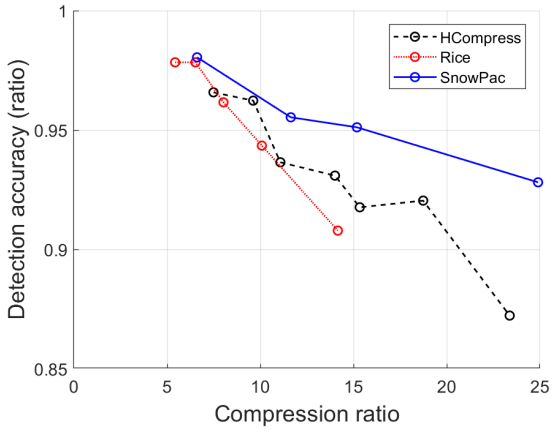

Figure 6. Detection accuracy vs compression ratio on DLS. Detection rates for our method are better preserved as we achieve higher levels of compression.

Compressor	CRatio	Total	Correct	Accuracy	FP
none	1.0	1433	–	–	–
HCompress	9.61	1487	1379	0.962	108
Rice	10.08	1471	1352	0.943	119
SnowPac	11.64	1453	1369	0.955	84

Table 3. DLS 10:1 compression ratio comparison. At near a 10:1 compression ratio, all methods begin to introduce noise resulting in more objects being detected incorrectly. From these compression types, our method resulted in the highest compression ratio while nearly matching HCompress in accuracy, and surpassing Rice in all metrics.

Alongside detection rates, we examine the effect on objects' R_mag ranges and their quantities in Figure 7. We map the distribution of objects throughout different R_mag ranges and observe the residual behavior. All methods perform similarly, showing that the total flux of each object is being preserved extremely well, as represented by the very small residuals. This behavior holds true up to a certain point; an R_mag of -10. After that point, much elevated amounts of residuals are observed for Rice and HCompress that represent larger deviations from the original R_mag distribution. Our method, SnowPac, is able to achieve some of the lowest residuals across all levels in latter R_mag ranges while achieving the highest compression ratio overall.

A per-object R_mag analysis is shown in Figure 8, where each object's original R_mag is compared as a function of the R_mag difference. All methods perform within a (+/−) 1 R_mag difference up to about an R_mag of -12, then objects and their quantities begin to deviate for some objects. Be-

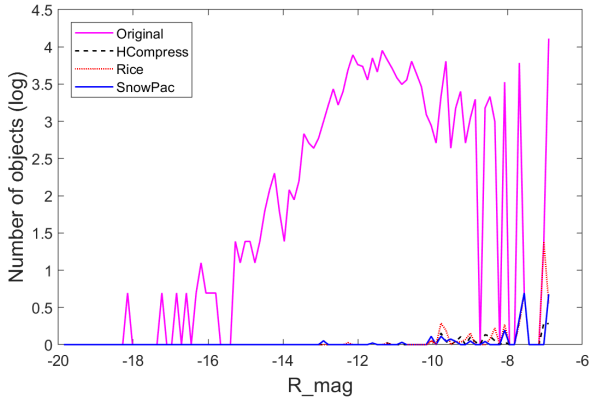

Figure 7. R_mag difference of compressed DLS data. The original R_mag quantity (magenta) is used to compare against the residual of lossy compression methods. SnowPac (blue) has the least amount of missing R_mag quantity for its detected objects compared to HCompress (black) and Rice (red). Bright and large objects are preserved well in general, but fainter objects classified by their R_mag are better preserved.

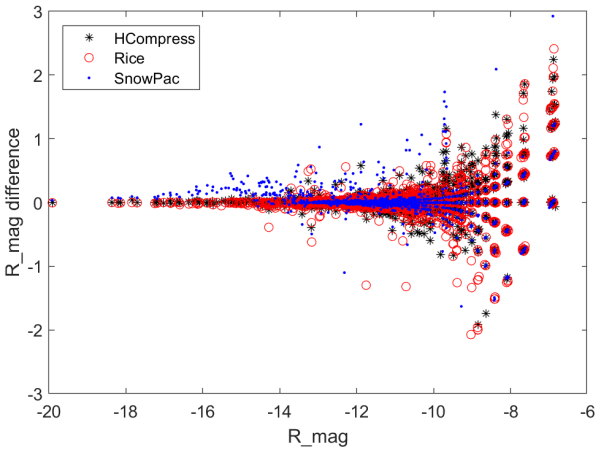

Figure 8. R_mag difference scatter plot for compressed DLS data. The original R_mag quantity is compared for each detected object and the difference is plotted. In general, all methods perform well up to R_mags of -12, where then quantities tend to deviate more than (+/-) 1. Notably, SnowPac has more objects with less error at larger ranges, but introduces a small amount of error in ranges between -17 to -13 compared to the other methods.

yond this range, our method has more objects clustered near zero signifying a better preservation of R_mag compared to the other methods. One point to note is the range between -17 and -13, where SnowPac has slightly more error than HCompress and Rice. Despite this small fluctuation in error, the detection results were not affected as observed in the rest of the analysis in this section.

To further evaluate the quality of lossy compression, three compression methods near the same compression ratio (10:1) are explored in derived object properties in Figures 9-11.

One important property of detected and classified objects is their ellipticity, which is used to distinguish stars from galaxies and must be preserved, along with the angle of the major axis, in order to properly measure weak gravitational lensing. Figure 9 compares the difference of the major axis in ellipticity and how it degrades. Lossy compression of medium and large objects shows to have minimal impact on the direction of the major axis. The clustering of objects towards the bottom of the y-axis shows that this property is preserved very well across different compression types. When evaluating the smallest objects, high levels of lossy compression may cause these objects that are oftentimes circular in nature to swap their major and minor axes. Although we have corrected this by comparing only the shortest difference between these two axes, the small and circular nature still prompts for larger shifts in direction for a small subset of the faintest objects. The clustering for the small objects towards the bottom-left shows a greater preservation of ellipticity for our method compared to HCompress and Rice.

Figure 10 derives the area for each detected object relative to the original dataset. Objects in the figure are sorted by the length of the major axis, meaning larger objects will be to the right of the plot. At a 10:1 compression ratio, areas of the largest objects start to have noticeable differences. While SnowPac has more objects with lower error, the maximums of the larger objects are impacted more than HCompress and Rice. This can be attributed to the underlying B-spline-based method that tends to erode high-frequency regions around objects. Larger objects seem to be the most susceptible to this but overall more objects with lower errors are achieved.

Figure 11 uses a flux-weighted, mean position for the area pixels to derive the center position of objects allowing for sub-pixel accuracy. At a 10:1 compression ratio, all objects are within 2 pixel accuracy to the original with the majority being within a single pixel accuracy and below. The distribution of object positions show two visible clusters of data points forming, those between a 10^{-2} to 10^{0} error and another between 10^{-3} and 10^{-2}. All three methods work within similar ranges in accuracy, but SnowPac achieves more objects with lower error (bottom) and has a tighter set of clustered points for slightly less accurate objects (top) compared to the rest. The tighter clustering of object errors showing low variance is preferred as it makes error more predictable. Finally, SnowPac is observed to have more objects with a smaller difference clearly visible by the clustering seen between a difference of 0.001 and 0.01 while achieving the highest compression ratio overall.

3.4 Evaluation – Dataset 2

The LSST dataset poses several more challenges due to its noisy, non-stacked nature but contains a lesser amount of astronomical objects per observable area. We compress the dataset at various compression ratios and show the results in Figure 12.

The LSST data is less object-dense when compared to the DLS due to its higher resolution and smaller sub-section of the sky. Objects also tend to be more elliptical. We extract a total of 401 objects and were able to preserve nearly 97.5% of the originally detected objects with a 10:1 compression ratio. Due to the data being less object-dense, we achieve higher compression results than HCompress and Rice; how-

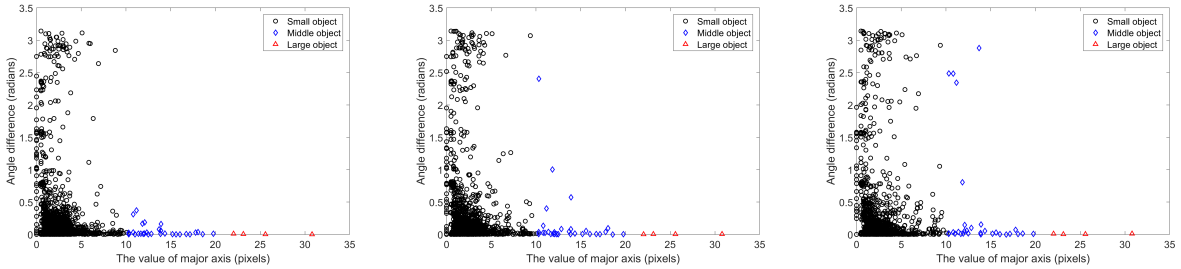

Figure 9. Object angle difference. From left to right: HCompress, Rice, and SnowPac on DLS data. Angle distributions exhibit clustering towards the bottom and left, indicating that many objects are detected with low error. Larger amounts of clustering for smaller objects can be observed for SnowPac, indicating a lower error compared to the rest. Large and Middle-sized objects have similar behavior across all compression methods.

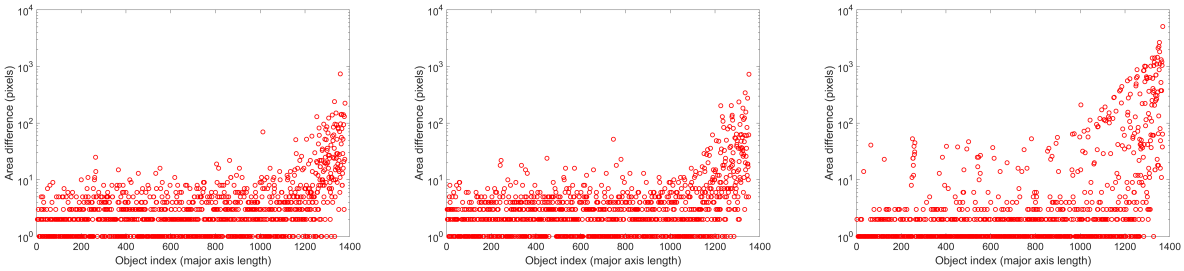

Figure 10. Object area difference. From left to right: HCompress, Rice, and SnowPac on DLS data. Sorted by the length of the major axis of objects, larger objects have their area most affected by lossy compression. While both HCompress and Rice have tighter bounds in area difference, there is more variance in preservation quality compared to SnowPac. While SnowPac has a few objects with higher residual area, a larger portion of objects are between 1 to 10 compared to the other methods.

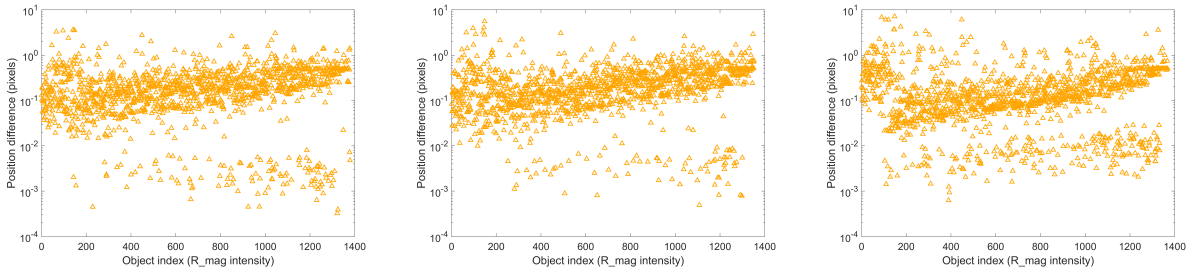

Figure 11. Object position difference. From left to right: HCompress, Rice, and SnowPac on DLS data. Sorted by R_mag intensity, most objects tend to have their center positions preserved pretty well. As sub-pixel accuracy is used, SnowPac is observed to have consistently lower difference amounts compared to the rest.

ever, lossy compression in general may compromise the future ability to stack multiple images to detect objects too faint to be detected on a single image. The datasets with input parameters closest to a 10:1 compression ratio were selected and analyzed starting in Tab. 4.

The highest compression ratio achievable by Rice was 6.86 and therefore used in this comparison. Both our method and HCompress achieved a near 10:1 ratio, with out method compressing slightly better. Despite having the best compression for this comparison, we achieve a 0.975 accuracy ratio when versus HCompress' 0.930 and Rice's 0.965.

Figure 13 compares the flux property of objects that were detected and their R_mag ranges. As shown, all of the methods are able to preserve flux well up to a certain range.

Compressor	CRatio	Total	Correct	Accuracy	FP
none	1.0	401	–	–	–
HCompress	9.63	374	373	0.930	1
Rice	6.86	389	387	0.965	2
SnowPac	9.79	396	391	0.975	4

Table 4. LSST 10:1 compression ratio comparison. The best compression possible by Rice is 6.86 therefore that is used. At near a 10:1 compression ratio, our method allows for higher number of correct objects, a higher accuracy ratio and better compression ratio.

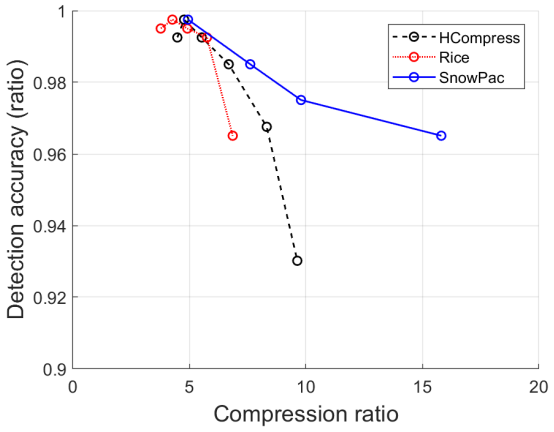

Figure 12. Detection accuracy vs. compression Ratio. The constant, sparse object nature of the LSST data allows our method to achieve higher detection rates and compression ratios. Other methods exhibit significant dropoffs in accuracy. The Rice compressor is unable to reduce the data any further.

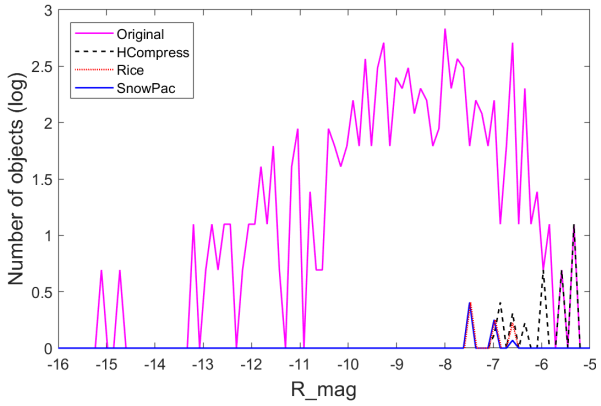

Figure 13. R_mag of compressed LSST data. The original R_mag quantity (magenta) is compared against the residual of various lossy compression methods. All methods are able to reproduce the original data well, but only SnowPac (blue) has the least amount of missing R_mag compared to HCompress (black) and Rice (red). Specifically, objects with the smallest flux are better preserved.

For ranges that correspond to some of the faintest objects, SnowPac is able to preserve the R_mag slightly better despite being able to achieve a higher compression ratio (9.79) compared against both Rice (6.86) and HCompress (9.63). The largest amounts of residual can be found for both of the other methods when they failed to detect objects in the -5.0 and -7.0 range. Effectively, objects in those flux ranges are completely lost while our method is able to preserve even at that scale.

For objects that are detected across all methods, Figure 14 plots each object's original R_mag as a function of the R_mag difference. Despite having the highest compression ratio overall, SnowPac shows to have the least amount

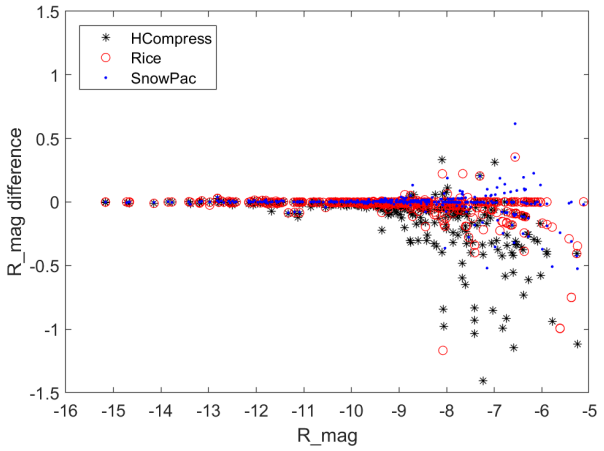

Figure 14. R_mag difference scatter plot for compressed LSST data. The original R_mag quantity is compared for each detected object and the difference is plotted. The distribution of delta R_mag show that SnowPac preserves this quantity more accurately than HCompress and Rice.

of delta R_mag compared to HCompress and Rice. There are a notable quantity objects with less error and the overall spread is less. This holds true despite Rice unable to compress beyond a ratio of 6.86, while our method reaches 9.79.

Additional insight can be gathered by analyzing the objects and their ellipticity. Figure 15 compares the quality of the major axis when computing the ellipticity of detected objects. In general, lossy compression of all types has little impact on the direction of the major axis signified by the clustering on the lower left. Nevertheless, the direction of the major axis is preserved very well for our method and Rice. SnowPac achieves similar results as Rice, despite having a larger compression ratio (9.79 vs 6.86). Additionally, when comparing SnowPac and HCompress with their similar compression ratios, SnowPac has tighter clustering for small object errors signifying a greater preservation of object properties.

Likewise, Figure 16 compares the area of each detected object relative to the original dataset. The index of objects are sorted by the length of the major axis, objects to the right of the figure are the largest in size. While achieving the highest compression ratio, our method is also able to preserve the physical properties of objects the best by having the lowest error in area computations.

Figure 17 analyzes the effects of lossy compression on deriving the center position of objects. The index of objects are sorted by their respective R_mag value, meaning left-to-right denotes fainter to brighter objects. In general, medium to high R_mag objects are affected the most by compression and the center-pixel computation of objects. At most, objects are 1.5 pixels off-center and the majority lie within sub-pixel accuracy. Similar to the DSL data, the LSST data shows an observable clustering behavior of two clusters of objects with error. Comparing HCompress and SnowPac, though both achieve similar compression ratios, SnowPac has more objects tightly clustered towards the upper regions and more numerous amounts of objects towards the bottom

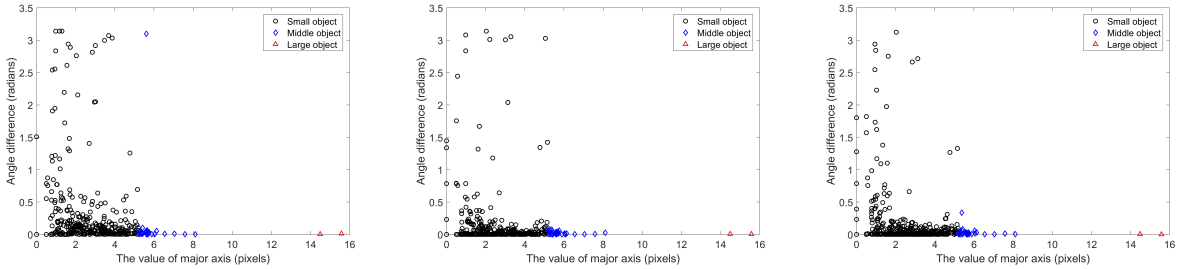

Figure 15. Object angle difference. From left to right: HCompress, Rice, and SnowPac on LSST data. Our method is able to achieve higher amounts of clustering towards the bottom portion of the angle-difference plot. SnowPac and Rice have similar amounts of error, despite having varying greatly in compression ratios.

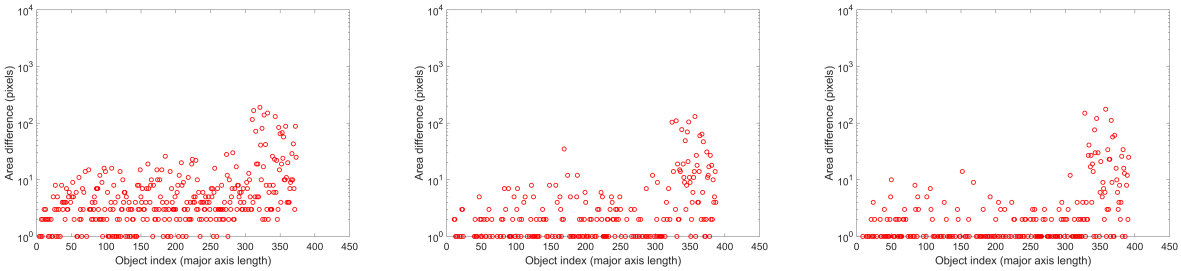

Figure 16. Object area difference. From left to right: HCompress, Rice, and SnowPac on LSST data. Sorted by the length of the major axis of objects, larger objects' areas are affected the most by lossy compression. Even while having the highest compression ratio, our method is able to preserve objects with higher precision having less impact on the object area computation.

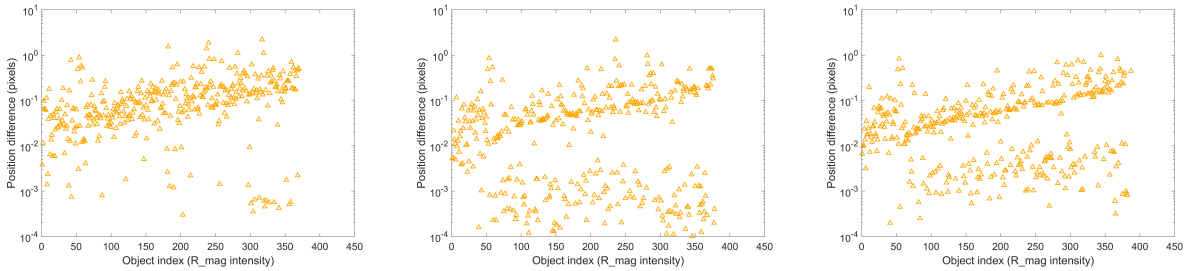

Figure 17. Object position difference. From left to right: HCompress, Rice, and SnowPac on LSST data. Sorted by the value of R_mag, most objects for all methods have a center computation error of at most 1.5 pixels. The majority of centers are within sub-pixel accuracy, with both SnowPac and Rice having the lowest. Rice has lower observable error only because it is unable to compress above a ratio of 6.86, compared to SnowPac's 9.79.

region. Although Rice is able to achieve more objects with lower error as seen by the clustering of object errors towards the lower regions, our method is able to nearly match Rice's detection performance with over an additional +3.0 compression ratio in savings.

3.5 Discussion

The selection of both types of data was made to test the effects of high and low density objects with lossy compression. While one dataset has a wider view of the sky with many more astronomical objects, the other has a narrower, more well defined series of objects at a lower count. The selection of these different ranges of data shows the flexibility of our compressor and its ability to preserve features at different density scenarios. Less object-dense datasets such as the LSST are able to achieve significantly higher ratios and accuracy compared to more dense datasets, highlighting one of the strengths of our compressor. Across several bands, we found that general compression performance was similar, which is expected, since it will adapt its top wavelet components to the PSF of the specific image being compressed.

We have selected input parameters for all methods to achieve extreme, medium, and low compression scenarios, shown in Figures 6 and 12. As shown in these sections, conservative compression ratios of about 5.0 on sparse-object datasets can achieve nearly an accuracy ratio of 0.99, and a compression ratio of about 10:1 an accuracy ratio of 0.975. The scaling of accuracy versus compression ratio makes our method the ideal choice for the general purpose compres-

sion of astronomical images. File sizes can be significantly reduced, much lower than standard Gzip by simply using higher quantization and the native wavelet hierarchy structure. As observed with DLS, a massive reduction can be made up to 25:1 compression ratio and still preserve 0.93 accuracy ratio of detected objects.

The compute tradeoff for using BZip2 vs Lz4 for encoding quantized coefficients may be significant in lower performing systems. BZip2, while extremely efficient in compression capabilities, is several magnitudes slower than Lz4. It is in this case that we would recommend using our method with Lz4 for high-performance applications. The general case, when file-size and compression ratio is the priority than BZip2 would suffice for general lossy compression of astronomical images.

4 CONCLUSIONS

Future sky survey telescopes will generate truly massive data sets, creating challenges for data representation, storage, transfer and analysis. Lossy data reduction methods, exceeding the compression capabilities of lossless compression methods, are necessary to reduce data sizes significantly. Lossy data compression has become more acceptable in many domains, and it is therefore crucial to understand what is lost when utilizing lossy methods. We have introduced and characterized a cubic B-spline wavelet-based lossy compression method that achieves high levels of data size reduction without significant loss of astronomically relevant objects.

Using qualitative and quantitative analysis, we have shown that SnowPac can conservatively preserve up to 98% of astronomical objects while achieving a 10:1 compression ratio, achieving better performance than both Rice and HCompress. This level of reduction is critical to support image-intensive data processing pipelines and minimize bottlenecks in integrated systems and network-based collaborative research. The method exhibits high fidelity with respect to original magnitudes and object shapes and orientations, which are critical for weak-lensing applications. The implementation of our method is freely available (Pulido 2019), in C++ and Matlab.

The current implementation uses fixed configurations of weights needed for quantization. This current implementation has made little effort to optimize the weights used for quantization, increasing the possibility of higher compression ratios in future versions. It is possible to make the software more flexible by having it explore image data for signs of structures of different scales to automatically adjust the weights, thereby permitting the adaptation of threshold values to the specific signals in an image.

Wavelet methods have also been studied for the purposes of cataloging and object-measuring performed directly on compressed data, since relevant object features are often captured in encoded form in specific scales of a wavelet representation. Taking advantage of this fact should lead to significant data processing acceleration, compared to data processing methods that can be applied only to decompressed images.

ACKNOWLEDGEMENTS

Jesus Pulido was supported in part by LANL and would like to thank them. Los Alamos National Laboratory is operated by Triad National Security, LLC for the US Department of Energy NNSA under contract no. 89233218NCA000001. Caixia Zheng was supported in part by the Fund of the Jilin Provincial Science and Technology Department under Grant Nos. 20190201305JC, 20180520215JH, 20180201089GX and 20170204018GX.

REFERENCES

Bertin, E. Arnouts, S. 1996, Astron. Astrophys. Suppl. Ser., 117, 393

Cohen A., Daubechies I., Feauveau J. C., 1992, Communications on Pure and Applied Mathematics, 45, 485

Collet Y., 2011, LZ4 - Extremely fast compression, http://lz4.github.io/lz4/

Daubechies I., 1992, PA: SIAM

Folk M., Heber G., Koziol Q., Pourmal E., Robinson D., 2011, in Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases. AD '11. ACM, New York, NY, USA, pp 36–47, doi:10.1145/1966895.1966900, http://doi.acm.org/10.1145/1966895.1966900

Fritze K., Lange M., Mostl G., Oleak H., Richter G. M., 1977, Astronomische Nachrichten, 298, 189

GNU 1997, GZIP, https://www.gnu.org/software/gzip/

Kitaeff V. V., Cannon A., Wicenec A., Taubman D., 2015, Astronomy and Computing, 12, 229

Kolev V., Tsvetkova K., Tsvetkov M., 2012, Publications of the Astronomical Society "Rudjer Boskovic", 11, 187

Masui K., et al., 2015, Astronomy and Computing, 12, 181

Morii M., Ikeda S., Sako S., Ohsawa R., 2017, The Astrophysical Journal, 835, 1

Pence W., 1999, in Mehringer D. M., Plante R. L., Roberts D. A., eds, Astronomical Society of the Pacific Conference Series Vol. 172, Astronomical Data Analysis Software and Systems VIII. p. 487

Pence W., 2009, Feasibility Study of using BZIP2 within the FITS Tiled Image Compression Convention, https://heasarc.gsfc.nasa.gov/fitsio/fpack/bzip2report.pdf

Pence W., Seaman R., White R., 2011, Fpack and Funpack User's Guide: FITS Image Compression Utilities, arXiv:1112.2671

Peters S. M., Kitaeff V. V., 2014, Astronomy and Computing, 6, 41

Price D. C., Barsdell B. R., Greenhill L. J., 2014, Is HDF5 a good format to replace UVFITS? (arXiv:1411.0507)

Pulido J., 2019, SNWPAC: SpliNe Wavelet Packing and Compression., https://github.com/lanl/VizAly-SNWPAC/

Pulido J., Livescu D., Woodring J., Ahrens J., Hamann B., 2016, Computers & Fluids, 125, 39

Rice R., Yeh P., Miller W. H., 1993, in Proceedings of AIAA Computing in Aerospace. AIAA, Reston, Virginia, USA

Shensa M. J., 1992, IEEE Trans. on Signal Processing, 40, 2464

Steward J., 1996, Bzip2 file compression, http://bzip.org

Vohl D., Fluke C. J., Vernardos G., 2015, Astronomy and Computing, 12, 200

Vohl D., Pritchard T., Andreoni I., Cooke J., Meade B., 2017, Publications of the Astronomical Society of Australia, 34

White R., Postman M., Lattanzi M., 1992, in MacGillivray H. T., Thompson E. B., eds, Digitized Optical Sky Surveys. p. 167

Wittman D. M., et al., 2002, in Tyson J. A., Wolff S., eds, Proc. SPIEVol. 4836, Survey and Other Telescope Technologies and Discoveries. pp 73–82 (arXiv:astro-ph/0210118), doi:10.1117/12.457348

Zheng C., Pulido J., Thorman P., Hamann B., 2015, Monthly Notices of the Royal Astronomical Society, 451, 4445

This paper has been typeset from a TEX/LATEX file prepared by the author.