Analyzing the Impact of Lossy Compressor Variability on Checkpointing Scientific Simulations

Pavlo Triantafyllides
Holcombe Department of Electrical and Computer Engineering
Clemson University
Clemson, South Carolina 29634
Email: ptriant@clemson.edu

Tasmia Reza
Holcombe Department of Electrical and Computer Engineering
Clemson University
Clemson, South Carolina 29634
Email: treza@clemson.edu

Jon C. Calhoun
Holcombe Department of Electrical and Computer Engineering
Clemson University
Clemson, South Carolina 29634
Email: jonccal@clemson.edu

***Abstract*—Lossy compression algorithms are effective tools to reduce the size of high-performance computing data sets. As established lossy compressors such as SZ and ZFP evolve, they seek to improve the compression/decompression bandwidth and the compression ratio. Algorithm improvements may alter the spatial distribution of errors in the compressed data even when using the same error bound and error bound type. If HPC applications are to compute on lossy compressed data, application users require an understanding of how the performance and spatial distribution of error changes. We explore how spatial distributions of error, compression/decompression bandwidth, and compression ratio change for HPC data sets from the applications PlasComCM and Nek5000 between various versions of SZ and ZFP. In addition, we explore how the spatial distribution of error impacts application correctness when restarting from lossy compressed checkpoints. We verify that known approaches to selecting error tolerances for lossy compressed checkpointing are robust to compressor selection and in the face of changes in the distribution of error.**

I. Introduction

High-performance computing (HPC) systems and applications have become central to rapid advancement in many fields of computational science and engineering, and can generate terabytes of data. Due to limitations on the available storage and I/O bandwidth on HPC systems, data reduction techniques — e.g., compression and decimation — are effective at reducing the volume of data written to the parallel file system. However, lossless compression approaches such as FPC [1], fp-zip [2], Gzip [3], and Zstd [4] cannot sufficiently reduce this volume for HPC use cases. On the other hand, lossy compression algorithms trade inaccuracies in the data for larger reductions in data size [5], [6].

Exascale systems will put further pressure on the memory system by increasing the total memory size of the system. In addition, these systems are expected to be characterized by lower mean-time between failures (MTBF) [7]. This puts extra pressure on the file system to handle more frequent checkpointing [8]. Current approaches to checkpoint-restart effectively utilize the complex memory hierarchy [9]–[12]. Nonetheless, employing data compression reduces checkpoint size and checkpointing time on all levels — i.e., local in-memory, neighbor, burst buffer, and parallel file system.

Using lossy compression for checkpoint-restart poses unique challenges for correctness. When a failure occurs that necessitates recovery from a checkpoint, error exists in the state variables that are used to advance the simulation. Therefore, the magnitude and distribution of error in these variables determines whether the restarted simulation advances correctly. Prior approaches use trial-and-error to determine what data can be lossy compressed and the acceptable error bounds [13]–[15] or establishes bounds based on the simulation's numerical accuracy [16], [17].

As development on lossy compression algorithms progresses, developers seek to improve the compression bandwidth, decompression bandwidth, and compression ratio of their compressors. In pursuit of these goals, the magnitude and distribution of error introduced in lossy compressed data can change. Understanding how lossy compressors change over time allows application users to understand how best to validate results when restarting from lossy compressed checkpoints (LCCs). If newer versions of a compressor yield compression errors that are identical to the prior versions, then little work is required to use the new version. However, if there are noticeable differences in the added compression errors, then revalidation of the lossy compression scheme may be required.

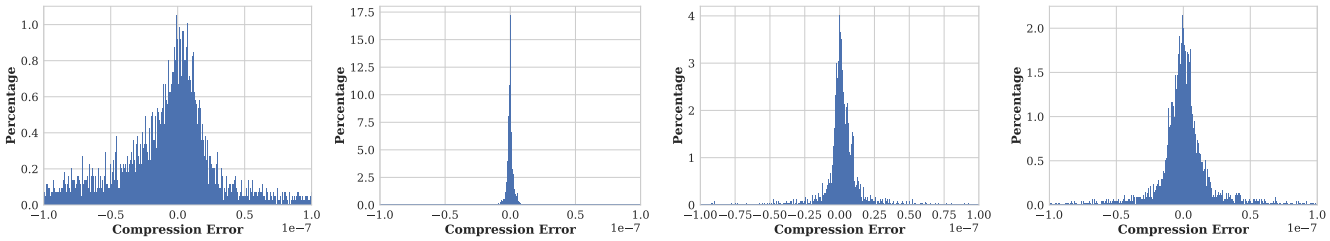
This paper investigates how variability of lossy compression algorithms impacts the ability to use lossy compression for checkpoint-restart. Our contributions are:

- validation of the robustness of prior work [16] that establishes bounds on acceptable compression error tolerances when considering changes in the compressor and different versions of the same lossy compressor;
- exploration of the variability of lossy compression algorithms across several generations with respect to performance and distribution of compression error;
- and analysis of the deviation in HPC data when restarting from lossy compressed checkpoints.

II. Motivation

Prior work [16] establishes a region of valid error tolerances to be used in lossy compressed checkpoint-restart (LCCR),

978-1-7281-4734-5/19/$31.00



(a) SZ-1.3 (b) SZ-1.4.11 (c) SZ-2.0 (d) ZFP-0.5.4

Fig. 1: Error distribution in x-component of momenta after evolving PlasComCM simulation for 20,540 time-steps after lossy restart.

Application	Problem	Checkpoint Size	Error Bound
PlasComCM [18]	2D homogeneous Euler flow past a fixed cylinder with non-periodic boundaries	4 MB	$1e{-}6$
Nek5000 [19]	Laminar 3D Navier-Stokes channel flow with periodic boundaries	16 MB	$1e{-}7$
Isabel [20]	Data set from Hurricane simulation	95 MB	$1e{-}6$

TABLE I: Test Applications

that upon restart do not yield error above the simulation's accuracy level dictated by the spatial discretization and choice of numerical method. The work claims the methodology is valid for any lossy compressor with absolute error bounding, but only presents results using SZ-1.3. Testing with additional compressors is required to verify its generality. Prior work [21] looks at error distribution produced by different lossy compressors after LCCR, but does not consider how the error distribution changes through successive iterations after recovery from a LCC.

Fig. 1 shows the distribution of compression errors in the x-component of momenta from PlasComCM [18] (see Section IV) at time-step 35,540. This figure shows that different versions of the same compressor can vary wildly in error distribution, indicating that generalizations made for an old version of a lossy compressor do not necessarily apply to newer versions. This error variation between versions of a single compressor requires additional analysis.

We validate the assertion made in [16] by testing the error propagation and attenuation for different versions of SZ and ZFP. We expand on [16] and [21] by considering how the error distribution changes after with each time-step after recovering two production applications used in prior work [16], PlasComCM [18] and Nek5000 [19], from LCCRs using each version of SZ and ZFP. In addition, we identify how the performance of each compressor improves and degrades across successive versions when compressing data from data sets of increasing size (see Table I).

III. Lossy Compression Algorithms

We test several versions of SZ and ZFP. We use SZ-1.3 because it is the last version of SZ that uses the original algorithm [5]; SZ-1.4.9b because it uses a new algorithm that compresses using multidimensional analysis [22]; SZ-1.4.11 because it includes an increase to the number of allowable data points; and SZ-2.0 because it uses a new algorithm that combines ideas from the previous algorithms [23]. We test with ZFP-0.4.1 because it is the oldest version of ZFP that obeys the current ZFP API and also works for 3D arrays that are not multiples of four; ZFP-0.5.1 because it adds an optimization to encode blocks with values of zero or values with magnitudes less than the specified tolerance using a single bit; and ZFP-0.5.4 because it is the latest and contains slight improvements for previously introduced features [24].

IV. Experimental Results

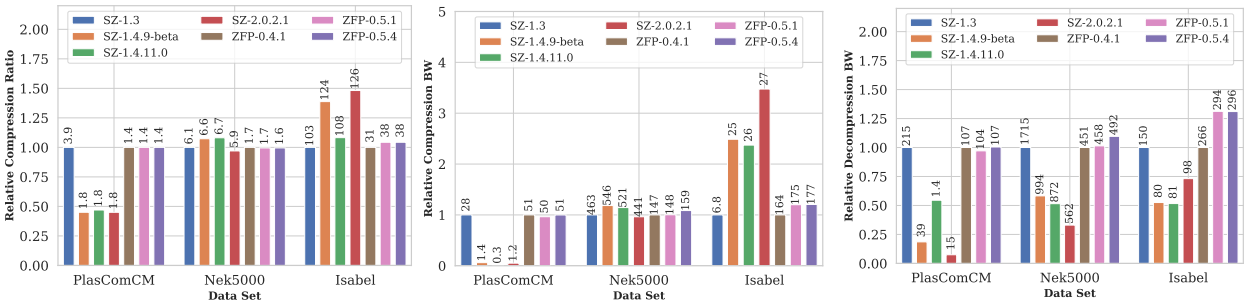
A. Testing Methodology

All experiments are run on Clemson's Palmetto Cluster [25] with nodes comprising: two Intel Xeon Gold 6148 CPUs at 2.4 GHz and 372 GB DDR4 RAM. GCC 4.8.1 compiles all lossy compressors and test applications in Table I. We use the Isabel data only to show impact of large data size. We select these test applications because they highlight error dissipation in PlasComCM and error retention with Nek5000 [16].

B. Impact on Compressor Performance

Newer releases of lossy compression algorithms add new features and improve performance. Users need to know how the newer versions perform in relation to the previous versions with respect to the compression ratio, compression bandwidth, and decompression bandwidth for effective integration.

Previous work [16] validates a compression error tolerance selection methodology using SZ-1.3 but argues that the methodology extends to any lossy compressor able to respect and absolute error bound. We provide further validation of this work by using SZ-1.3 and ZFP-0.4.1 as baselines to show how newer versions of SZ and ZFP perform on our test applications (see Table I). Thus, we explore the impact of the compressors on small, medium, and large data sets. To analyze how newer versions function as replacements for already validated workflows, we set up our configuration files as similarly as possible to the baseline.



(a) Compression Ratio. (b) Compression Bandwidth (MB/s). (c) Decompression Bandwidth (MB/s).

Fig. 2: Relative performance of newer versions of SZ and ZFP to their baseline. Raw averages shown above lines bars.

In order to determine how performance metrics such as compression ratio and compression and decompression bandwidth change across versions, we collect data across 2550 time-steps from PlascomCM, 300 time-steps from Nek5000, and one time-step of the hurricane Isabel data set. We compress and decompress the time-step data using each compressor with an absolute error bound. When compressing the data sets in Table I, we use established bounds from literature [16], [20]. Averaging each metric across all variables for each compressor version results in each version having a single value for each metric at each time-step. Normalizing at each time-step against the corresponding metric and time-step from the baseline compressor, then averaging across all time-steps determines if performance improves with newer versions of each compressor. We compute the averages in this way because the compression ratios for the PlascomCM and Isabel data sets vary by orders of magnitude between variables.

Fig. 2 shows that as the data size increases, newer versions of SZ and ZFP show an increase in performance over their baselines, except for the decompression bandwidth for newer versions of SZ. Although the decompression bandwidth for newer versions of SZ is no better than the baseline in our tests, its performance relative to the baseline does slightly improve across successive versions. The newer ZFP versions perform consistently for the PlascomCM and Nek5000 data sets and show slight improvements in relative performance as the data size increases. Decompression bandwidth benefits the most for new versions of ZFP.

SZ is more sensitive to changes in data size, where as ZFP shows more stable performance across data sizes. For small data sets, the newer SZ versions perform worse than the baseline and should not be used in those circumstances. Newer versions of SZ are better suited to large data sets than previous versions. This highlights SZ's design focus on targeting large data volumes that bottleneck large-scale systems, as seen in the recent algorithm changes. The algorithm underlying ZFP has been more stable, as indicated by more consistent performance. Newer versions that better compress special cases — e.g., such as blocks of all zeros — help ZFP perform better on larger data sets that leverage more of these optimizations.

While the newer versions of SZ show lower decompression bandwidth, they provide better compression and significantly better compression bandwidth for large data sets than the baseline. This indicates that these versions are much better for checkpointing or storing for analysis/archiving large data sets. Because the decompression bandwidth is lower than the SZ-1.3 baseline, recovery from a lossy compressed checkpoint is a greater burden with newer versions of SZ. ZFP's consistent compression and decompression bandwidth irrespective of file size combined with its amenability to hardware implementation [6] indicate that it is well suited for memory compression and network communication compression.

We conclude that ZFP offers more stability that SZ. However, as the algorithm in SZ stabilizes, performance variation should stabilize as well.

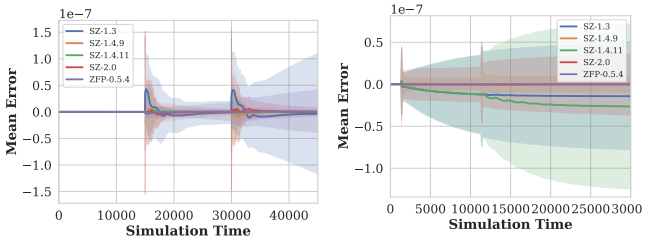
C. Impact on HPC Data

Prior work [16] verifies correctness by computing the max-norm between the state variables and ensuring it is less than the simulation's level of accuracy. This analysis shows that there are differences in how the simulation responds to the compression error upon restart. However, the max-norm is only a point-wise measure of deviation in the state variables and does not account for how common this deviation is. Investigating deviation in more detail offers insight as to how statistical properties of the simulation may change when restarting from an LCC.

1) PlasComCM

Fig. 3(a) shows the evolution of mean error for the x-component of momenta in PlasComCM (other variables exhibit similar behavior). There is no deviation in the state variables before restart at time-step 15,000. Over time, as erroneous flow leaves the domain due to non-periodic boundary conditions, the mean errors move toward zero. Until this occurs, statistics computed with this data are perturbed. Restarting from the second LCC at time 30,000 causes similar mean errors that reduce over time. The magnitude of error in all the variables is always less than the simulation's accuracy of $1.8e{-}5$. Thus, yielding numerically equivalent runs to a simulation using lossless compression [16].

Investigating the standard deviation in the mean error shows noticeable differences between SZ and ZFP and between different version of SZ. For example, SZ-1.3 produces the



(a) PlasComCM: x-momenta.

(b) Nek5000: x-velocity.

Fig. 3: Mean error in select variables from between PlasComCM and Nek5000 simulations restarting from lossy compressed checkpoints compared to simulations restarting from normal checkpoints. The shaded region about the mean indicates the standard deviation.

largest spread in distribution of compression error. Newer versions of SZ greatly improve on this spread. Over time, the standard deviation in the compression error for newer versions of SZ reduces while SZ-1.3's standard deviation increases. The best SZ version with respect to mean error and standard deviation is SZ-1.4.9. ZFP behaves most similarly to SZ-1.3 as the standard deviation in mean error grows over time.

2) Nek5000

Fig. 3(b) shows the evolution of the average error for the x-component of velocity for Nek5000. As with PlasComCM, there is no deviation in the state variables before the first restart at time-step 1,500. At this point the mean error deviates from zero and remains there for the remainder of the simulation. After the second restart at time-step 11,500, the mean error deviates further, and is most evident in SZ-1.4.11. Comparing the mean error for the other SZ versions to SZ-1.3, SZ-1.4.9 performs the best. Because the magnitude of error in the simulation checkpointed using ZFP is noticeably lower than that in the simulations checkpointed using versions of SZ, ZFP outperforms SZ. For all variables, the magnitude of error is always less than the simulation's accuracy of $1e{-}5$. Thus, yielding numerically equivalent runs to a simulation using lossless compression [16].

Examining the standard deviation of mean error in Fig. 3 highlights notable physical properties of the simulation, namely periodic boundary conditions and direction of flow. For all state variables and compressors, the standard deviation increases with successive restarts and remains nearly constant as the simulation evolves. This indicates that the error does not leaving the domain due to the selection of periodic boundary conditions. For this simulation, the fluid flows in the x direction. Because this variable has a large range, it has a unique distribution of error. The y and z-components (not shown) of velocity have similar distributions of values and therefore have similar distributions of error. The pressure variable's values (not shown) are fairly uniform which leads to a low standard deviation in the mean error.

D. Discussion

Comparing the distribution of errors in the PlasComCM and Nek5000 simulations highlight the importance of physical properties on the attenuation and propagation of compression error. When using LCCR with simulations using periodic boundary conditions, simulations must ensure that persistent deviations do not impact accuracy with repeated restarts.

Although not required in these simulations, conservation laws would not be persevered as the mean error for all the state variables differs from zero. This conveys the need for further exploration of how statistical properties change when advancing the simulation with lossy compressed data.

V. Related Work

Recent work explores the error distribution of various lossy compression algorithms when compressing HPC data and shows that the distribution of compression errors noticeably changes between compressors [21]. Our work explores how the variability in the error distribution changes between different versions of the same compressor and how this change impacts correctness of simulations that use lossy compressed checkpoint-restart. Analyzing the performance of lossy compression algorithms helps create performance models and define use cases [16], [26]–[28]. Prior approaches explore trial-and-error approaches for selecting the compressor's error bound for checkpoint-restart [13], [14], in-line computation [15], or data analytics [29]–[31]. Other work explores how different error bounding metrics impact floating-point truncation error [32] and the resulting distribution of compression error [21] and how to define error tolerance selection methodologies for checkpointing [16], [17].

VI. Conclusion

Prior work [16] details an error tolerance selection methodology, claiming that it is robust to selection of any lossy compressor with absolute error bounding. We further explore its validity, by restarting PlascomCM [18] and Nek5000 [19] from LCCs generated by several versions of SZ and ZFP, and analyzing how the error distribution changes across time steps. When considering the distribution of errors, newer versions of SZ improve over SZ-1.3 by having a lower standard deviation in the mean error of the state variables. ZFP yields almost identical distribution of errors, which allows for easy transition to newer versions without workflow revalidation. Because the error stays well within tolerance for all time-steps, we conclude that the methodology is robust to selection of lossy compressor. In addition, our experimental results show that new versions of the SZ compressor yield better performance on larger data sets, sacrificing performance on smaller data sets. New versions of ZFP generally perform better than previous versions in all metrics when compressing data sets of any size.

Acknowledgements

Hurricane Isabel data produced by the Weather Research and Forecast (WRF) model, courtesy of NCAR, and the U.S. National Science Foundation (NSF). This material is based upon work supported by the National Science Foundation under Grant No. SHF-1910197.

REFERENCES

[1] M. Burtscher and P. Ratanaworabhan, "High throughput compression of double-precision floating-point data," in *Data Compression Conference, 2007. DCC '07*, March 2007, pp. 293–302.

[2] P. Lindstrom and M. Isenburg, "Fast and efficient compression of floating-point data," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 12, no. 5, pp. 1245–1250, 2006. [Online]. Available: http://dx.doi.org/10.1109/tvcg.2006.143

[3] P. Deutsch, "Gzip file format specification version 4.3," United States, Tech. Rep., 1996.

[4] Y. Collet and M. Kucherawy, "Zstandard Compression and the application/zstd Media Type," RFC 8478, Oct. 2018. [Online]. Available: https://rfc-editor.org/rfc/rfc8478.txt

[5] S. Di and F. Cappello, "Fast error-bounded lossy HPC data compression with SZ," in *2016 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2016, Chicago, IL, USA, May 23-27, 2016*, 2016, pp. 730–739. [Online]. Available: https://doi.org/10.1109/IPDPS.2016.11

[6] P. Lindstrom, "Fixed-rate compressed floating-point arrays," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2674–2683, Dec 2014.

[7] M. Snir, R. W. Wisniewski, J. A. Abraham, S. V. Adve, S. Bagchi, P. Balaji, J. Belak, P. Bose, F. Cappello, B. Carlson, A. A. Chien, P. Coteus, N. A. DeBardeleben, P. C. Diniz, C. Engelmann, M. Erez, S. Fazzari, A. Geist, R. Gupta, F. Johnson, S. Krishnamoorthy, S. Leyffer, D. Liberty, S. Mitra, T. Munson, R. Schreiber, J. Stearley, and E. V. Hensbergen, "Addressing failures in exascale computing," *International Journal of High Performance Computing Applications*, vol. 28, no. 2, pp. 127 – 171, May 2014.

[8] F. Cappello, A. Geist, B. Gropp, L. Kale, B. Kramer, and M. Snir, "Toward exascale resilience," *Int. J. High Perform. Comput. Appl.*, vol. 23, no. 4, pp. 374–388, Nov. 2009. [Online]. Available: http://dx.doi.org/10.1177/1094342009347767

[9] L. Bautista-Gomez, S. Tsuboi, D. Komatitsch, F. Cappello, N. Maruyama, and S. Matsuoka, "FTI: high performance fault tolerance interface for hybrid systems," in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '11. New York, NY, USA: ACM, 2011, pp. 32:1–32:32. [Online]. Available: http://doi.acm.org/10.1145/2063384.2063427

[10] A. Moody, G. Bronevetsky, K. Mohror, and B. R. d. Supinski, "Design, modeling, and evaluation of a scalable multi-level checkpointing system," in *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 1–11. [Online]. Available: http://dx.doi.org/10.1109/SC.2010.18

[11] "Very-low overhead checkpointing system," https://github.com/ecp-veloc/veloc, accessed: July 25, 2018.

[12] N. Liu, J. Cope, P. H. Carns, C. D. Carothers, R. B. Ross, G. Grider, A. Crume, and C. Maltzahn, "On the role of burst buffers in leadership-class storage systems," in *IEEE 28th Symposium on Mass Storage Systems and Technologies, MSST 2012, April 16-20, 2012, Asilomar Conference Grounds, Pacific Grove, CA, USA*, 2012, pp. 1–11. [Online]. Available: http://dx.doi.org/10.1109/MSST.2012.6232369

[13] X. Ni, T. Islam, K. Mohror, A. Moody, and L. V. Kale, "Lossy compression for checkpointing: Fallible or feasible?" in *Poster Session of the 2014 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '14. Washington, DC, USA: IEEE Computer Society, 2014.

[14] N. Sasaki, K. Sato, T. Endo, and S. Matsuoka, "Exploration of lossy compression for application-level checkpoint/restart," in *Proceedings of the 2015 IEEE International Parallel and Distributed Processing Symposium*, ser. IPDPS '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 914–922. [Online]. Available: http://dx.doi.org/10.1109/IPDPS.2015.67

[15] D. Laney, S. Langer, C. Weber, P. Lindstrom, and A. Wegener, "Assessing the effects of data compression in simulations using physically motivated metrics," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '13. New York, NY, USA: ACM, 2013, pp. 76:1–76:12. [Online]. Available: http://doi.acm.org/10.1145/2503210.2503283

[16] J. Calhoun, F. Cappello, L. N. Olson, M. Snir, and W. D. Gropp, "Exploring the feasibility of lossy compression for pde simulations," *The International Journal of High Performance Computing Applications*, vol. 33, no. 2, pp. 397–410, 2019. [Online]. Available: https://doi.org/10.1177/1094342018762036

[17] D. Tao, S. Di, X. Liang, Z. Chen, and F. Cappello, "Improving performance of iterative methods by lossy checkponting," in *Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing*, ser. HPDC '18. New York, NY, USA: ACM, 2018, pp. 52–65. [Online]. Available: http://doi.acm.org/10.1145/3208040.3208050

[18] "Plascomcm," https://bitbucket.org/xpacc-dev/plascomcm.

[19] J. W. L. Paul F. Fischer and S. G. Kerkemeier, "nek5000 Web page," 2019, http://nek5000.mcs.anl.gov.

[20] "Ieee visualization 2004 contest: Data set," http://vis.computer.org/vis2004contest/data.html, accessed: May 22, 2019.

[21] P. Lindstrom, "Error distributions of lossy floating-point compressors," *Joint Statistical Meetings 2017*, pp. 2574–2589, October 2017.

[22] D. Tao, S. Di, Z. Chen, and F. Cappello, "Significantly improving lossy compression for scientific data sets based on multidimensional prediction and error-controlled quantization," in *2017 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2017, Orlando, FL, USA, May 29 - June 2, 2017*, 2017, pp. 1129–1139. [Online]. Available: https://doi.org/10.1109/IPDPS.2017.115

[23] X. Liang, S. Di, D. Tao, S. Li, S. Li, H. Guo, Z. Chen, and F. Cappello, "Error-controlled lossy compression optimized for high compression ratios of scientific datasets," in *IEEE International Conference on Big Data, Big Data 2018, Seattle, WA, USA, December 10-13, 2018*, 2018, pp. 438–447. [Online]. Available: https://doi.org/10.1109/BigData.2018.8622520

[24] "zfp versions | computation," https://computation.llnl.gov/projects/floating-point-compression/zfp-versions.

[25] "Palmetto cluster documentation," https://www.palmetto.clemson.edu/palmetto/userguide_palmetto_overview.html.

[26] D. Ibtesham, K. B. Ferreira, and D. C. Arnold, "A checkpoint compression study for high-performance computing systems," *IJHPCA*, vol. 29, no. 4, pp. 387–402, 2015. [Online]. Available: http://dx.doi.org/10.1177/1094342015570921

[27] S. W. Son, Z. Chen, W. Hendrix, A. Agrawal, W. keng Liao, and A. Choudhary, "Data compression for the exascale computing era - survey," *Supercomputing frontiers and innovations*, vol. 1, no. 2, 2014. [Online]. Available: http://superfri.org/superfri/article/view/13

[28] T. Lu, Q. Liu, X. He, H. Luo, E. Suchyta, J. Choi, N. Podhorszki, S. Klasky, M. Wolf, T. Liu, and Z. Qiao, "Understanding and modeling lossy compression schemes on HPC scientific data," in *IPDPS*. IEEE Computer Society, 2018, pp. 348–357.

[29] A. H. Baker, H. Xu, J. M. Dennis, M. N. Levy, D. Nychka, S. A. Mickelson, J. Edwards, M. Vertenstein, and A. Wegener, "A methodology for evaluating the impact of data compression on climate simulation data," in *Proceedings of the 23rd International Symposium on High-performance Parallel and Distributed Computing*, ser. HPDC '14. New York, NY, USA: ACM, 2014, pp. 203–214. [Online]. Available: http://doi.acm.org/10.1145/2600212.2600217

[30] I. Foster, M. Ainsworth, B. Allen, J. Bessac, F. Cappello, J. Y. Choi, E. Constantinescu, P. E. Davis, S. Di, W. Di, H. Guo, S. Klasky, K. K. Van Dam, T. Kurc, Q. Liu, A. Malik, K. Mehta, K. Mueller, T. Munson, G. Ostouchov, M. Parashar, T. Peterka, L. Pouchard, D. Tao, O. Tugluk, S. Wild, M. Wolf, J. M. Wozniak, W. Xu, and S. Yoo, "Computing just what you need: Online data analysis and reduction at extreme scales," in *Euro-Par 2017: Parallel Processing*, F. F. Rivera, T. F. Pena, and J. C. Cabaleiro, Eds. Cham: Springer International Publishing, 2017, pp. 3–19.

[31] J. Nardi, N. Feldman, A. Poppick, A. Baker, and D. Hammerling, "Statistical analysis of compressed climate data," NCAR, Tech. Rep., 2018.

[32] J. Diffenderfer, A. Fox, J. Hittinger, G. Sanders, and P. Lindstrom, "Error analysis of zfp compression for floating-point data," *SIAM Journal on Scientific Computing*, 02 2019.