ROOT Files Improved with Extreme Compression

J. Gonzalez, J. Lauret (PI)
G. Van Buren, M. Burtscher,
I.A. Cali, Ph. Canal, R. Nunez, Y. Ying

ACAT 2021, Daejeon, South Korea

Compression in ROOT

- The ***ROOT*** framework has compression capabilities included since day 1
  - Initially started with ZLIB algorithm, new algorithms added if and only if they bring significant improvement: smaller files with LZMA, faster decompression with LZ4.
- ***lossy*** compression ***algorithms*** heavily used in image/sound processing but have yet to be applied in Physics (except for case by case hand coded data preconditioning)
- For the past 7 years, Accelogic pioneered and perfected a radically new theory of numerical computing codenamed "Compressive Computing".
- If seamlessly integrated into ***ROOT***, it would lower the bar of acceptance as most HENP experiments are using ***ROOT*** formats for their output workflows.

ROOT
Data Analysis Framework

accelogic

About accelogic compression

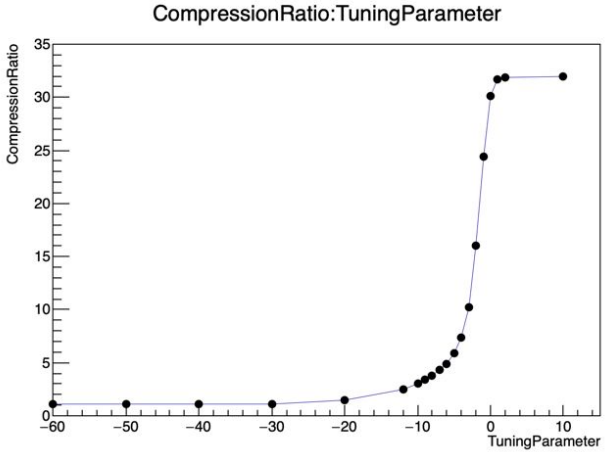
- Accelogic was founded in 2005 and targets the groundbreaking acceleration of any software through novel, easily-injectable techniques. This work is part of a DOE/NP SBIR program.

- Accelogic offers lossless and lossy compression algorithms
  - lossless still provide stunning compression values

- Accelogic's "lossy" compression techniques idea - argument can be made that if bits carry zero or insignificant information, then losing their content is not a true "loss"
  - Reasons for this: limit of the measurement, precision, insignificant value (do we care of the gram level for measuring the weight of an elephant?), ...
  - Codes provided by Accelogic offer tunable parameters controlling the level of lossiness
  - For now, this must be evaluated to find the optimal compression without information loss

About testing accelogic compression

- We performed considerable testing *outside of* ***ROOT*** [on binary files extracted from ***ROOT*** tree] data and *inside-of-root* [“*root2root*” conversion]
- Tests used ***convoluted Physics signals*** (i.e. not a single variable distribution)
- Data volume reduction: Compression ratio
- Compression and decompression speed
- Tests across: Different data types and Different buffer sizes



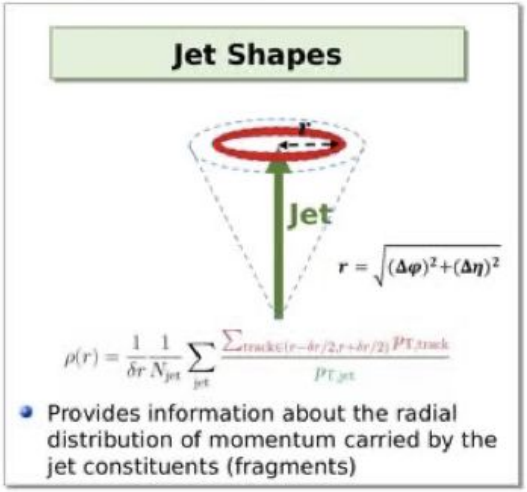
The tuning parameter indicates the compression strength. Our tuning knob varies from -60 to 10 (illustration based on STAR’s data, similar for CMS)

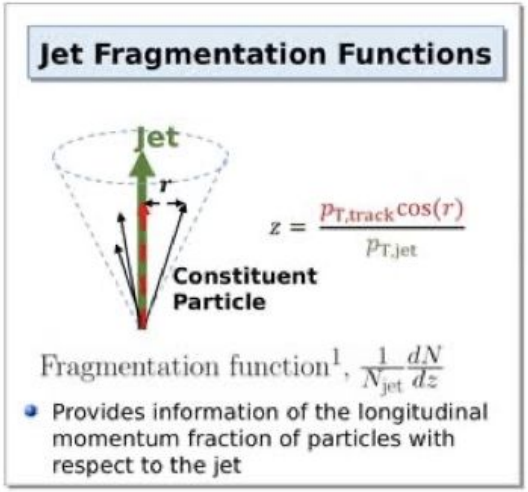
ROOT Integration and its challenges

- Added new compression engine (prefix ‘B’/’L’)
  - BLAST library in active development to include more of Accelogic’s compression schemes
- Lossy Compression is entirely type dependent
  - Need to pass the Branch/Basket data type down to the compression engine.
- Basket’s content is byteswapped (on x86)
  - Compression engine needs to byteswap it back to understand it
- Required use of `ROOT::Experimental::EIOFeatures::kGenerateOffsetMap`
  - Otherwise buffer has a mix of data types and the trailing 32-bit integers describing array lengths or entry boundaries are destroyed by the lossy floating compression (who treat them as float)
- Usage is as usual (call to SetCompressionSettings/Algorithm/Level)
  - Algorithm: ROOT::EAlgorithm::kBLAST
  - If data type is not supported, currently return with no compression (*fallback to be improved later)*
- Compression ***within ROOT*** compares well with ***outside-of-ROOT***
  - Some plots shown here are ***outside-of-ROOT***

CMS Experiment study: jets

- CMS team has started with exploring *outside-of-ROOT compression* on selectcolumns of data:
    - `Entry number - int`
    - `Particle ID - int`
    - `Status code - int`
    - `pT - double`
    - `Eta - double`
    - `Phi - double`
    - `Mass - double`





- Workflow:
    - Extracting values from the original ***ROOT*** tree → write to binary file → use compression algo on binary files → compute tracks' momentum from decoded values → run jet clustering algorithm

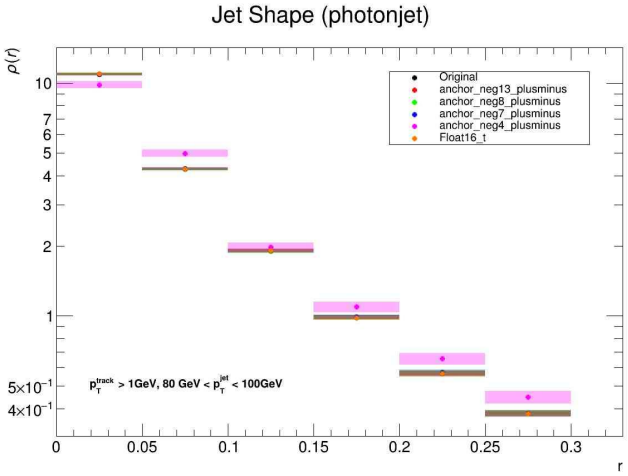
Jet Substructure

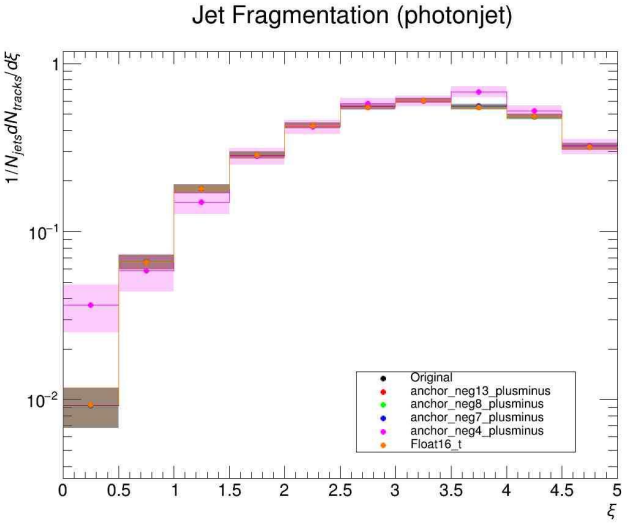
- **Jet shape**: radial distribution of momentum of particles
- **Jet fragmentation**: longitudinal momentum fraction of particles w.r.t. jet
- x = {7, 8, 13} parameters look ok (further analysis required for conclusive understanding); x = 4 appears too lossy and not viable
- @ x=13, Accelogic compression ratios range from ~ 4-15 while gzip would provide ~ 2-4 . From x2 to x4 improvement!

Compression Ratios

	13	8	7	4	gzip	float16
pT	4.25	6.24	6.88	9.95	1.97	2
Eta	3.75	5.27	5.75	7.86	1.95	2
Phi	4.15	6.04	6.65	9.54	2.02	2
Mass	14.95	17.25	18.12	22.02	3.69	2
Overall	**6.13**	**8.32**	**8.98**	**11.81**	**2.95**	**2.63**

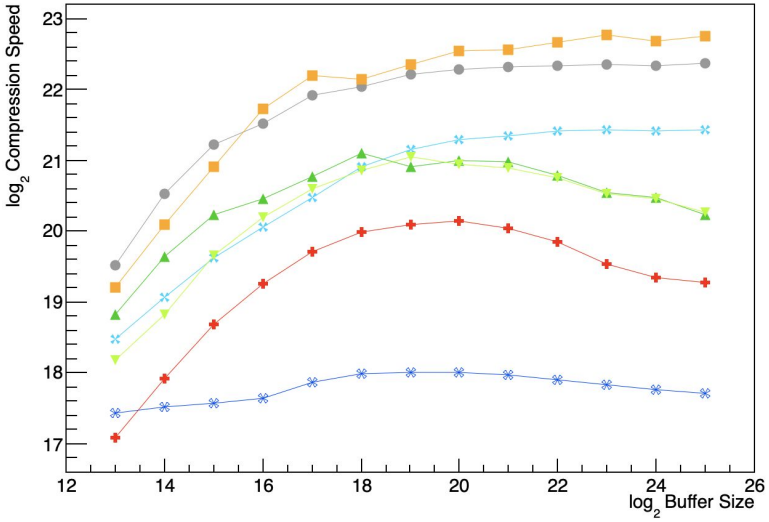
(Overall includes Accelogic RLE int compression)

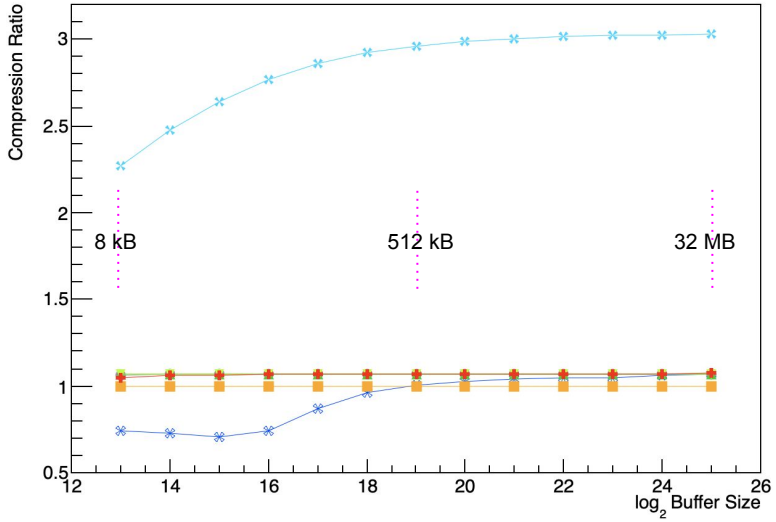


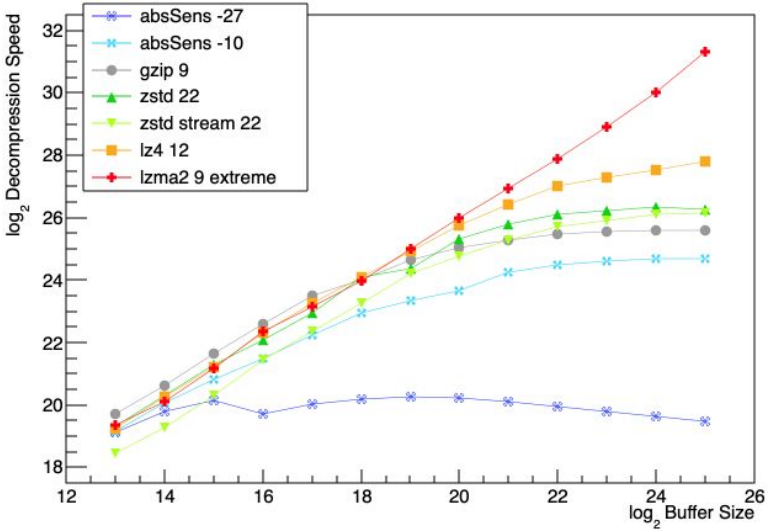


Decompression Speed

- Showing highest compression setting of each lossless encoder
- Showing ***Accelogic*** compression of STAR data with no noticeable changes at all (x = -27), and with no/negligible loss of information (x = -10)

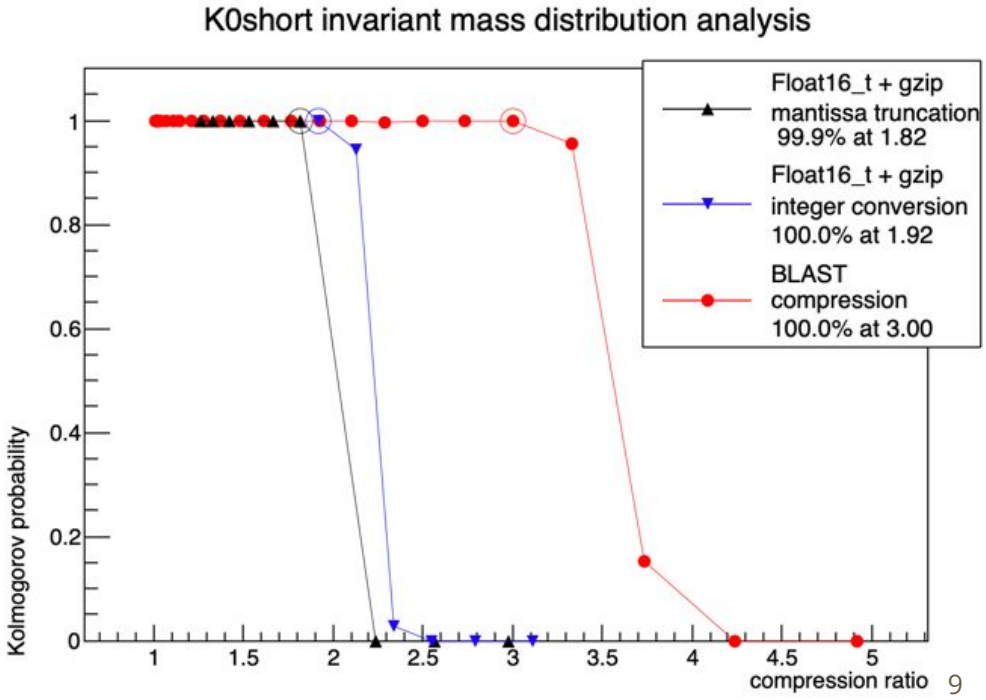






Other Lossy Compression Techniques

- ***ROOT*** offers Float16_t, Double32_t
- Or users pack & encode to reduce bits, e.g. 32-bit float => 16-bit short
- Can still be combined with subsequent lossless compression
- Example from ***STAR*** data K0s reconstruction with compression of momentum components:



Open issues and next steps

- ***Patent***: Accelogic is VERY interested and committed in making the release and granting of all licenses immediately after project closing (June 2022).
  - This would include free unlimited permission for ***ROOT*** to use and integrate both the patents and the codes (i.e. redistribute sources as part of ***ROOT*** releases)
  - Patent process seems to be on schedule so far.
  - In the interim, binary library distribution are available for early adopter.
- Ability to save "***lost***" bits in separate file (***ZIG***) is not yet available.
  - It is next on our plate and believed to be key to build end-user's ease and confidence in using lossy compressions
- Instrumentations for automatic discovery the optimal value not provided for now …
  - Subtleties of compression and data organization in experiment specific format may require performing hopefully just *once* an analysis similar to the work from STAR and CMS mentioned here.
- **Key finding** - both CMS and STAR (two independent experiments with different analysis data formats) tend to find that the **same range of lossiness configuration settings** lead to a negligible loss of information

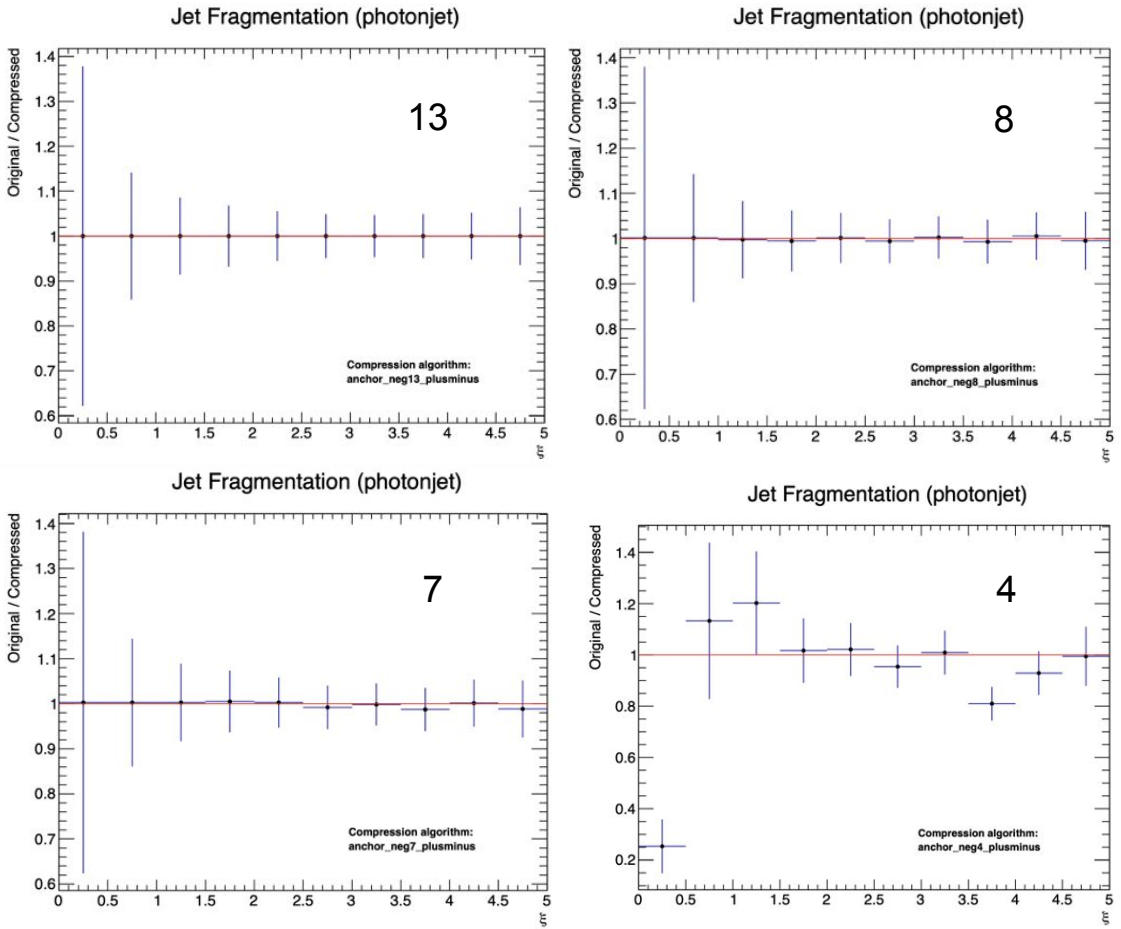
Conclusions

- Ready for distribution soon (more progress by ACAT, wrap up in next few months) - binary lib distribution at first
- Compression factors exceeds those of classic compression algorithm by x2 to x4 (i.e. overall compression factors > x4+ but < 9)
  - Early “outside ***ROOT***” result confirmed “inside ***ROOT*** file”
- Benefits drop when compared to Float16 / Double32 but still better (more gain and can be used without class declaration changes)

BACKUP SLIDES

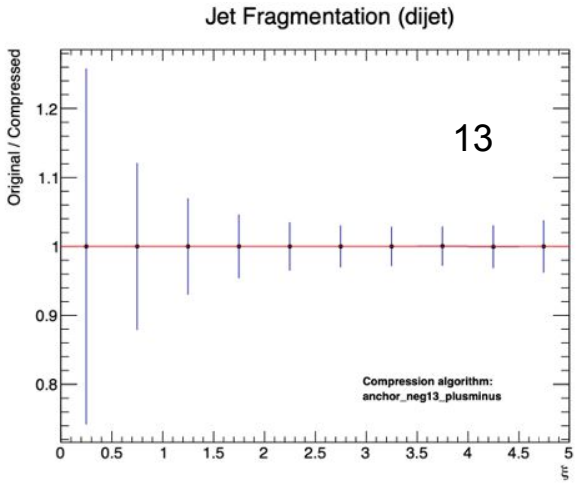
Jet Fragmentation Ratios

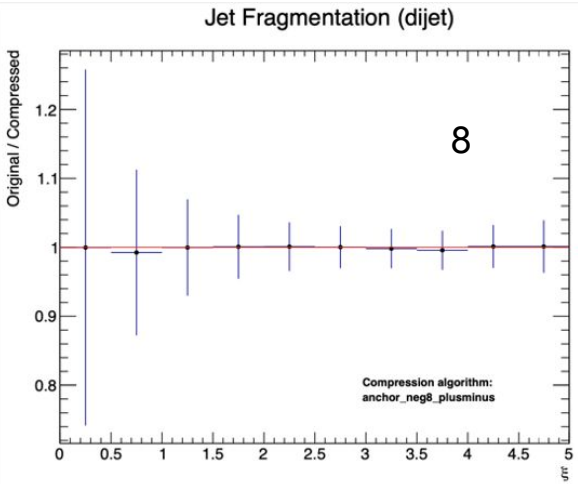
- Photon-jet sample
- 13: matches exactly
- 7, 8: appear to match very well
- 4: unacceptable

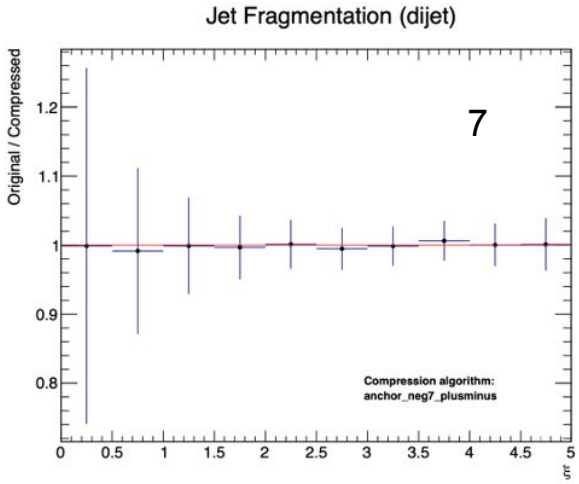


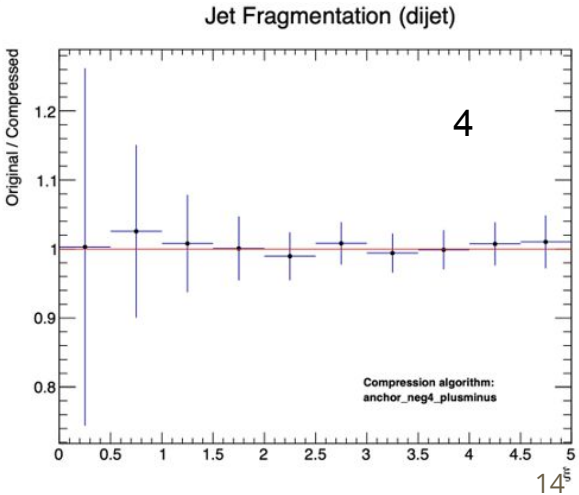
Jet Fragmentation Ratios

- Dijet sample
- 13: matches exactly
- 7, 8: match quite well
- 4: bin 2 seems to have higher error but actually quite good here





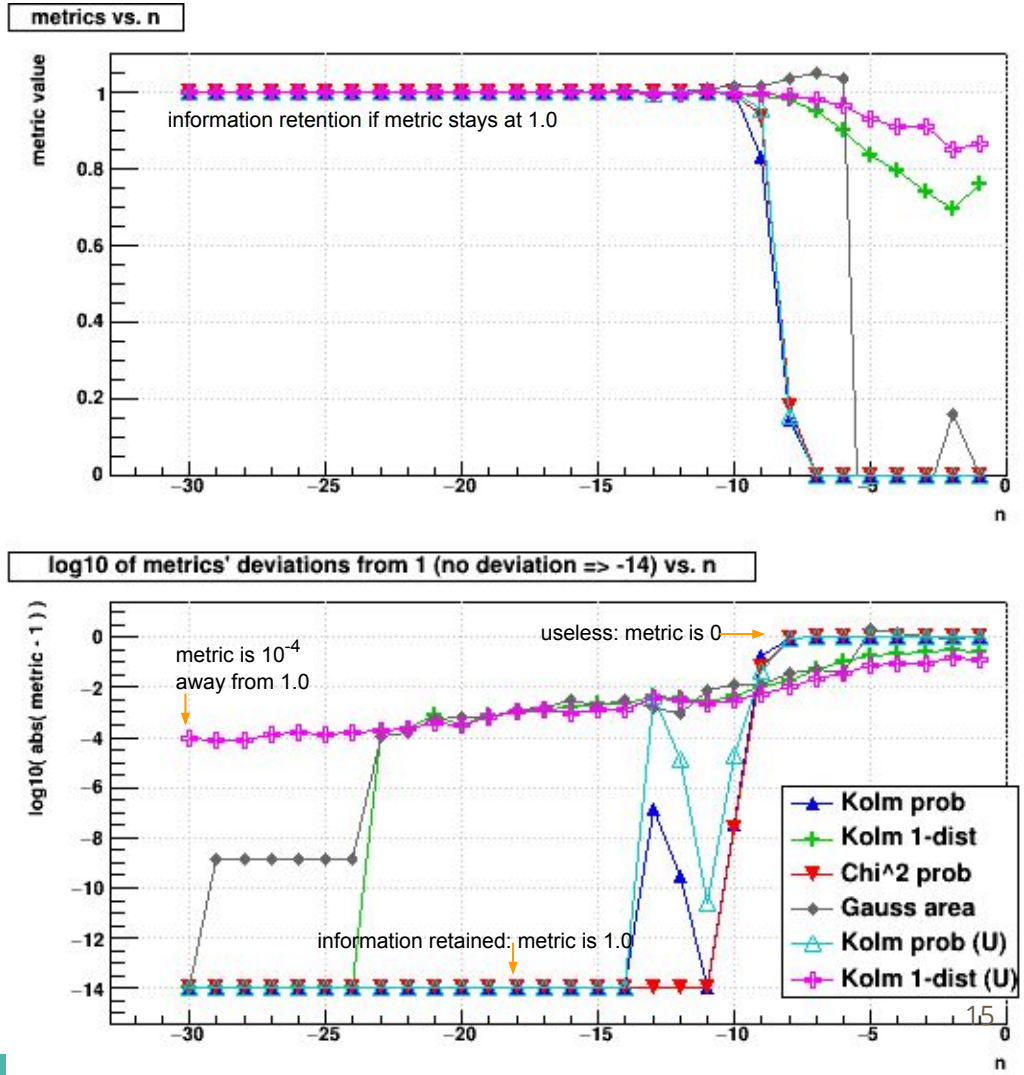




Compression inside ROOT (~2 MB buffer)

Evolution of information retention/loss metrics

- 6 metrics selected which should be 1.0 for retained information:
  - Kolmogorov probability that binned histograms are the same shape, and unbinned distributions are compatible, (U) label
  - 1 minus the Kolmogorov “maximum distance” for binned & unbinned (U)
  - Chi^2 probability that binned histograms are the same shape
  - normalized **Gaussian area** from the (background+signal) fit
- Lower plot revisualizes same information, as $\log_{10}$ of deviation from desired value of 1.0



Buffer Size: Decompression Speed vs. Compression Ratio

- Plot shows only a small range of the comp ratios in a region of interest
- Colors represent Buffer Size (small in blues, large in reds)
- Bands indicated by hand-drawn dashed-orange curves are for constant “n”
- Trends are very similar for compression speeds for this range of “n” and comp ratios
- **Conclusion**: Buffer sizes below ~512 kB (or perhaps ~1 MB) lose rapidly in speed, but gradually in comp ratio
  (8 kB ~7x slower, but ~1.3x less compression than 1 - 32 MB)