ADIOS for scientific data

Scott Klasky: ORNL/GT/UTK

April 11, 2018

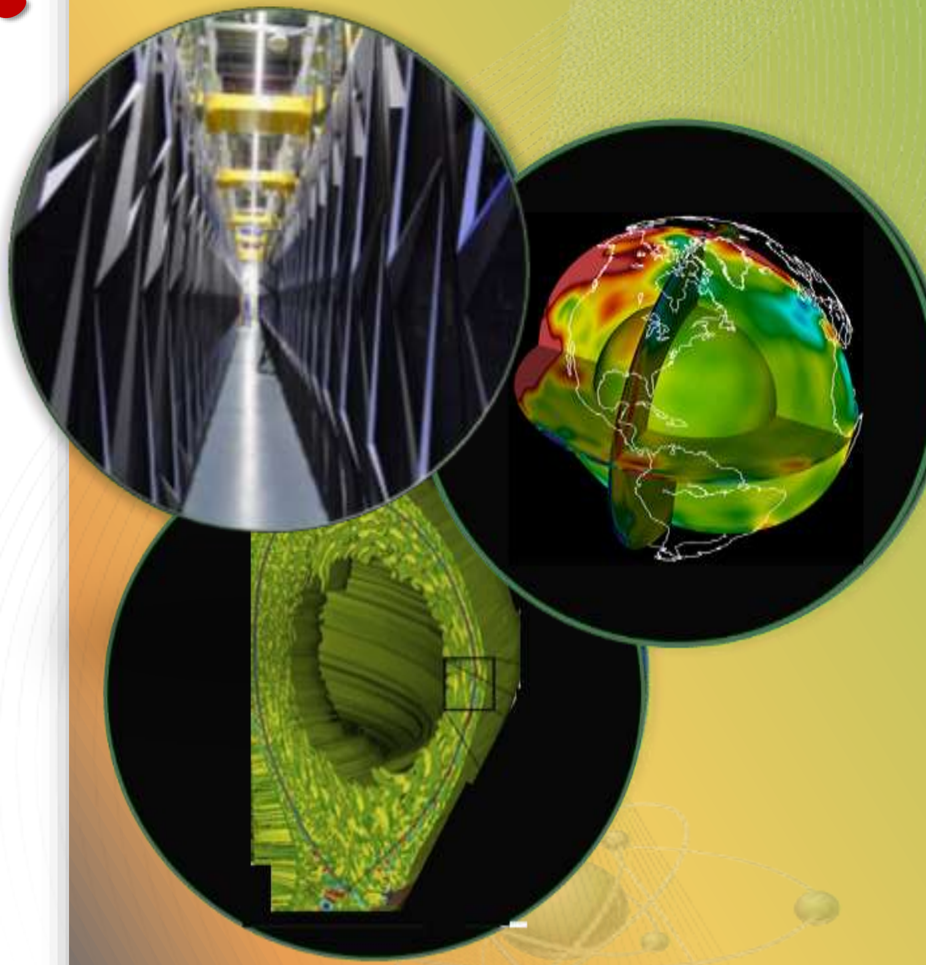
PSI

https://github.com/ornladios/ADIOS

https://github.com/ornladios/ADIOS2

Scientific Data Group
Matthew Wolf (Deputy)
Scientific Data Management
Norbert Podhorszki –TL
Mark Ainsworth
Jong Choi
William Godoy
Tahsin Kurc
Qing Liu
Jeremy Logan
Kshitij Mehta
Eric Suchyta
Ruonan Wan
Jason Wang
Scientific Data Analytics
Dave Pugmire – TL
Mark Kim
James Kress
George Ostrouchov

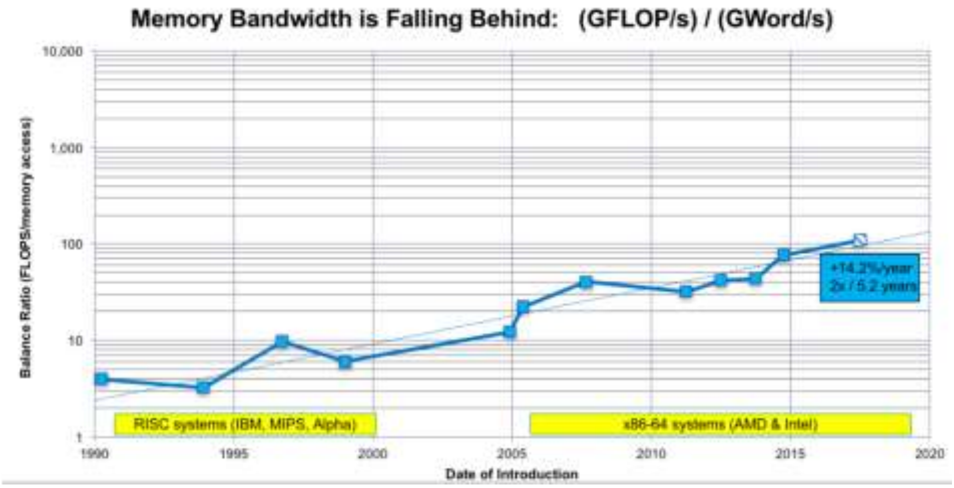
``

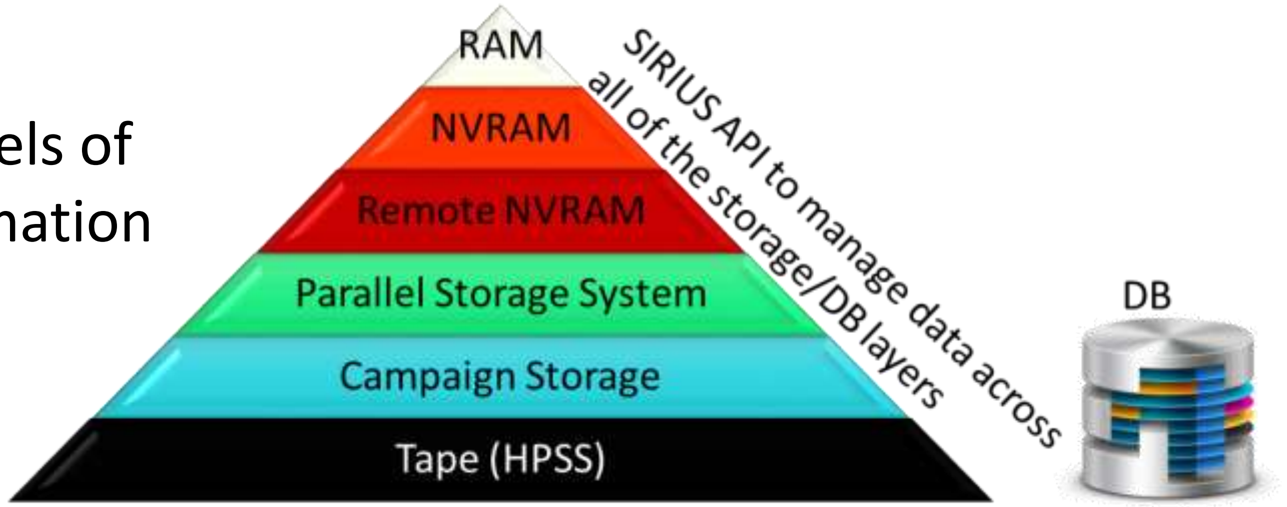


Georgia Tech , Rutgers, Kitware, ParaTools, HDF, PPPL, Sandia, LBNL, ANL, BNL, Oregon, Rutgers,, ++

I/O on HPC machines is challenging

- Problem
  - File system/network bandwidth does not keep up with computing power
  - Too much data which is written to the storage system is either purged or never read back for post-processing
- Approach
  - Refactor the data into different levels of *importance* according to the information content.





To solve many of these challenges we created ADIOS

Category	Goal	Paper (citations, award)
WAN I/O + viz	Reduce time to solution	S. Klasky, S. Ethier, Z. Lin, K. Martins, D. McCune, R. Samtaney, Grid-based parallel data streaming implemented for the gyrokinetic toroidal code in **SC 2003**. (75)
Checkpoints	Save the state at the end of the job (large output)	J. Lofstead, F. Zheng, S. Klasky, K. Schwan, Adaptable, metadata rich IO methods for portable high performance IO in **IPDPS 2009** (159)
Variability	Reduce the I/O variability	J. Lofstead, F. Zheng, Q. Liu, S. Klasky, et al., Managing variability in the IO performance of petascale storage systems in **SC 2010** (129)
In transit	Create I/O staging for HPC	Abbasi, M. Wolf, G. S. Eisenhauer, S. A. Klasky, K. Schwan, F. Zheng, DataStager: Scalable Data Staging Services for Petascale applications. **Cluster 2010**. (224)
In transit analytics	In situ workflows for analytics	F. Zheng, H. Abbasi, C. Docan, J. Lofstead, Q. Liu, S. Klasky, M. Parashar, N. Podhorszki, K. Schwan, M. Wolf, PreDatA–preparatory data analytics on peta-scale machines in **IPDPS 2010** (162)
Reading	Reading patterns	J. Lofstead, M. Polte, G. Gibson, S. Klasky, K. Schwan, R. Oldfield, M. Wolf, Q. Liu, Six degrees of scientific data: reading patterns for extreme scale science IO in **HPDC 2011**.(75)
Data queries	Queries + reduction	S. Lakshminarasimhan, J. Jenkins, I. Arkatkar, Z. Gong, H. Kolla, S.-H. Ku, S. Ethier, J. Chen, C.-S. Chang, S. Klasky, et al., ISABELA-QA: query-driven analytics with ISABELAcompressed extreme-scale scientific data in **SC 2011**. (67)
Lossy Compression	Reduce output size	S. Lakshminarasimhan, N. Shah, S. Ethier, S. Klasky, et al., Compressing the incompressible with ISABELA: In-situ reduction of spatio-temporal data in Euro-Par **2011**. (99)

To solve many of these challenges we created ADIOS

Category	Goal	Paper
Hybrid staging for viz	Combine in situ + in transit	J. C. Bennett, H. Abbasi, P.-T. Bremer, R. Grout, A. Gyulassy, T. Jin, S. Klasky, H. Kolla, M. Parashar, V. Pascucci, et al., Combining in-situ and in-transit processing to enable extreme-scale scientific analysis in **SC, 2012** **(130)**
Code Coupling	XSOA	C. Docan, M. Parashar, S. Klasky. DataSpaces: an interaction and coordination framework for coupled simulation workflows. **Cluster** 2012 **(151)**
Hybrid staging infrastructure	Resource sharing on nodes	F. Zheng, H. Yu, C. Hantas, M. Wolf, G. Eisenhauer, K. Schwan, H. Abbasi, S. Klasky, GoldRush: Resource Efficient In Situ Scientific Data Analytics Using Fine-Grained Interference Aware Execution in SC **2013** **(50)**
Diagnostics	Small but frequent output	Q. Liu, J. Logan, Y. Tian, H. Abbasi, N. Podhorszki, J. Y. Choi, S. Klasky, R. Tchoua, et al.. Hello adios: the challenges and lessons of developing leadership class i/o frameworks. *Concurrency and Computation: Practice and Experience* **2014**, *26*, 1453–1473. **(74)**
Modeling Variability	Understand variability	L. Wan, M. Wolf, F. Wang, J.Choi, G. Ostrouchov, S. Klasky, Analysis and Modeling of the End-to-End I/O Performance on OLCF's Titan Supercomputer, HPCC **2017**, **Best Paper Nominee.**
Understand reduction	Understand impact of reduction to errors	Tao Lu, Qing Liu, Xubin He, Huizhang Luo, Eric Suchyta, Norbert Podhorszki, Scott Klasky, Matthew Wolf, Tong Liu, Understanding and Modeling Lossy Compression Schemes on HPC Scientific Data, **IPDPS 18**, **best paper nominee.**
Queries	Optimize querying of large scientific data	K. Wu, J. Gu, S. Klasky, N. Podhorszki, J. Qiang, "Querying Large Scientific Data Sets with Adaptble IO System ADIOS", SC Asia **2018.**, **outstanding Technical Paper Award.**
Multilevel data reduction	Reduce + quantify data reduction	M. Ainsworth, O. Tugluk, B. Whitney, S. Klasky, "Multilevel Techniques for Compression and Reduction of Scientific Data --The Multivariate Case", **SIAM** Journal on Scientific Computing, Submitted for publication **2018**.

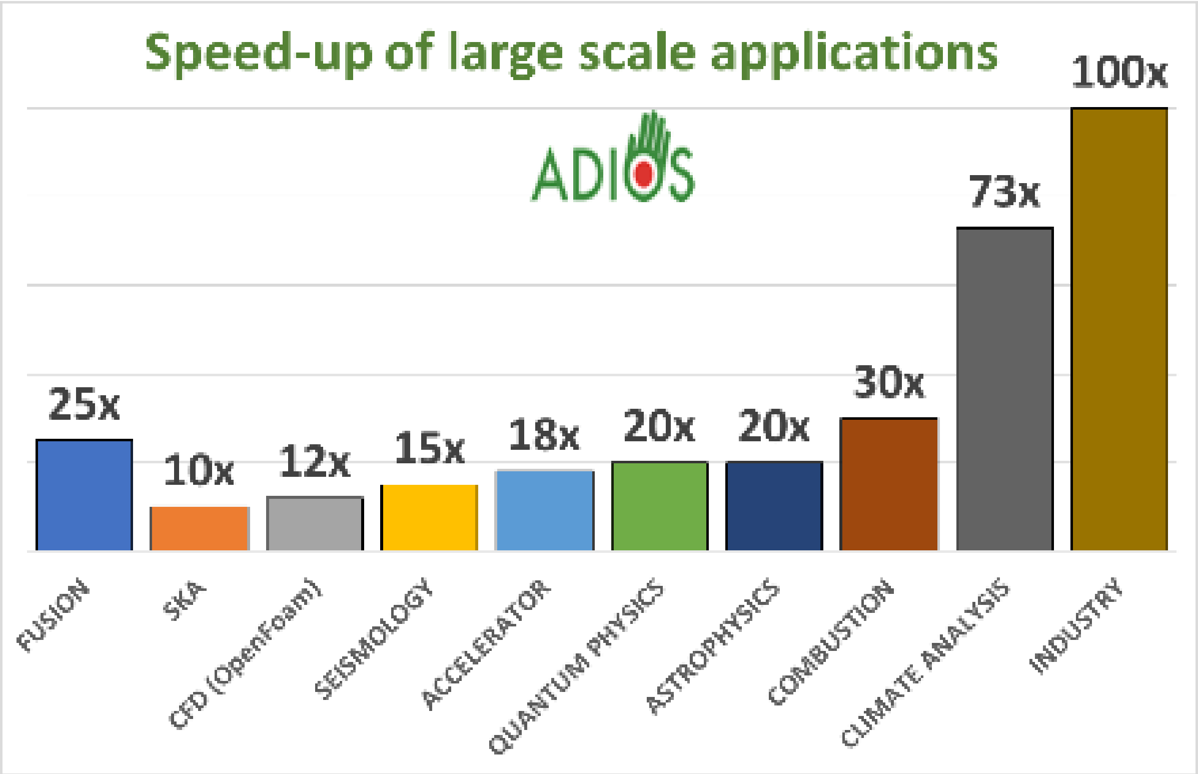
What is ADIOS

- An extendable framework that allows developers to *plug-in*
  - **I/O methods**: Aggregate, Posix, MPI
  - **Services**: Compression, Decompression
  - **File Formats**: HDF5, netcdf, …
  - **Stream** Format: ADIOS-BP
  - **Plug-ins**: Analytic, Visualization
  - **Indexing**: FastBit, ISABELLA-QA
- Incorporates the "best" practices in the I/O middleware layer
- Incorporates self describing data streams and files
  - https://www.olcf.ornl.gov/center-projects/adios/, https://github.com/ornladios/ADIOS
- Available at ALCF, OLCF, NERSC, CSCS, Tianhe-1,2 Pawsey SC, Ostrava

- ADIOS core – provides the basic infrastructure
  - BP stream format, Memory Buffering, Data Movement strategies
- ADIOS library - allow "best practice" from external components
  - Engines, Transformations, Indexing, Transports
- ADIOS Framework – allow scientific libraries to be used inside ADIOS
  - Staging libraries, reduction libraries, Indexing libraries, I/O libraries
- ADIOS ecosystem – Allow applications to interact with ADIOS codes/data
  - Analysis- Visualization services, Performance services, Living Miniapps

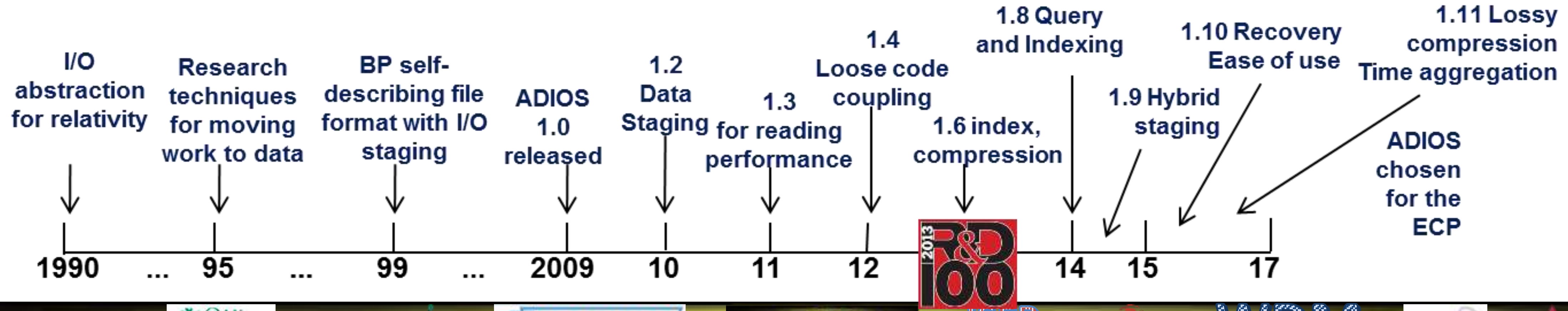
Q. Liu, J. Logan, Y. Tian, H. Abbasi, N. Podhorszki, J. Y. Choi, S. Klasky, R. Tchoua, J. Lofstead, R. Oldfield, et al.. Hello adios: the challenges and lessons of developing leadership class i/o frameworks. *Concurrency and Computation: Practice and Experience* **2014**, *26*, 1453–1473.

I/O Framework for Data Intensive Science



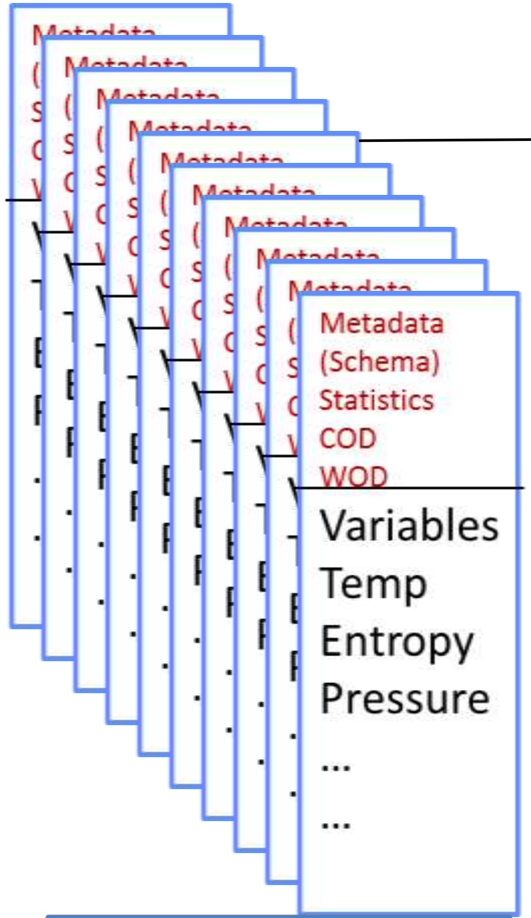
ADIOS Collaborating Institutions

ANL
Auburn University
BNL
Brown University
Chinese Academy of Sciences
CMU
Delaware University
Duke
Emory University
Georgia Tech University
HZDR
KAUST
KISTI
Kitware
LANL
LBNL
LLNL
U. Maryland
U. of Mainz
NASA
NJIT
North Carolina State University
NREL
NWU
ORNL (other groups)
Peeking University
PPPL
Princeton University
Rutgers University
SNL
Stanford University
Stony Brook
Tokyo Tech University
Tsinghua University
U. C. Davis
U. C. Irvine
U. Tenn. Knoxville
U. Texas at Austin
U. Utah
U. Western Australia



klasky@ornl.gov

Create a next-generation file/stream format



Ensemble of chunks = file

- All data chunks are from a single producer
  - MPI process, Single diagnostic
- Ability to create a separate metadata file when "sub-files" are generated
- Allows variables to be individually compressed
- Has a schema to introspect the information
- Has workflows embedded into the data streams
- Format is for "data-in-motion" and "data-at-rest"
- Log-like data format

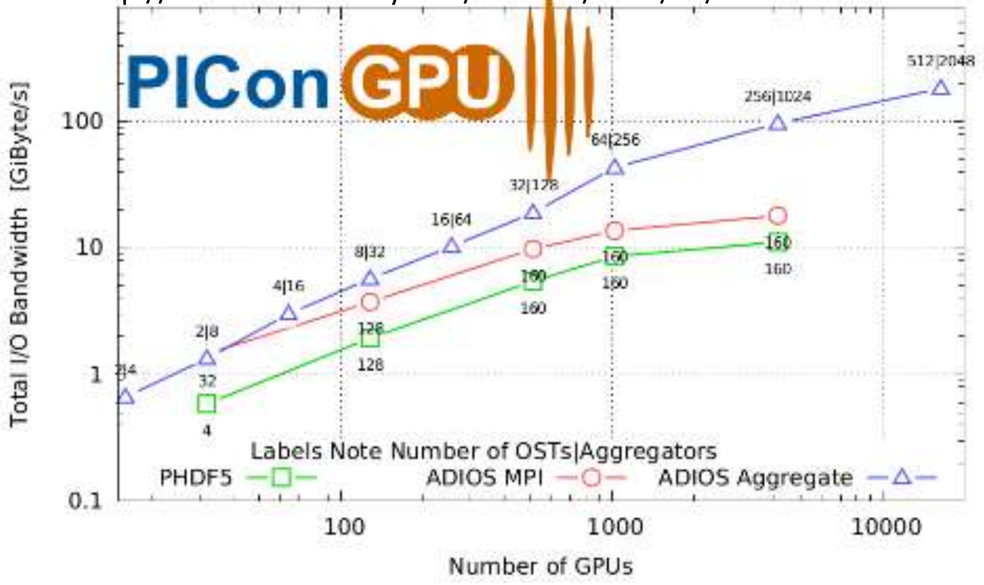
J. Lofstead, F. Zheng, S. Klasky, K. Schwan, *Adaptable, metadata rich IO methods for portable high performance IO* in *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*, IEEE, pp. 1–10.

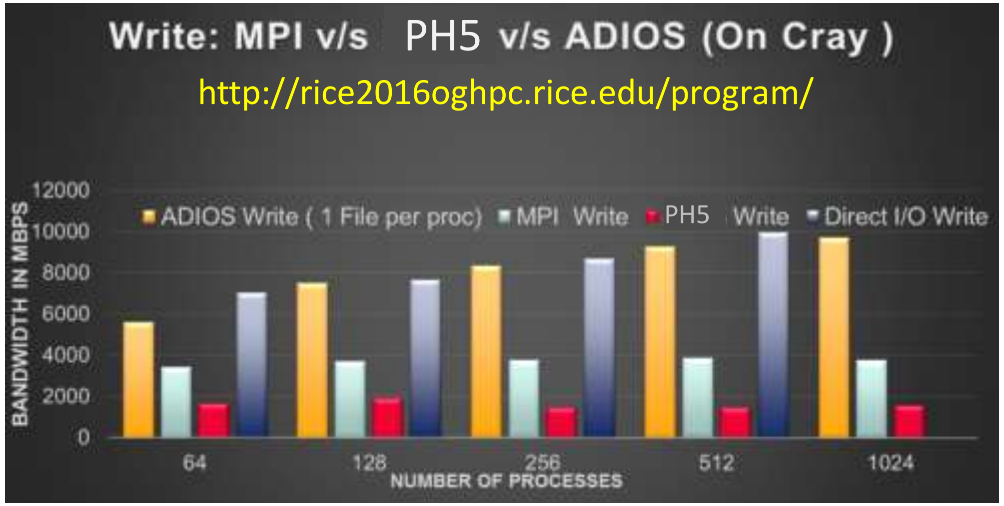
klasky@ornl.gov

Impact to LCF applications

- Accelerators – PIConGPU
  - M. Bussmann, et al. - HZDR
  - Study laser-driven acceleration of ion beams and its use for therapy of cancer
  - Computational laboratory for real-time processing for optimizing parameters of the laser
  - Over 200 GB/s on 16K nodes on Titan
- Seismic Imaging – RTM by Total Inc.
  - Pierre-Yves Aquilanti, TOTAL E&P in context of a CRADA
  - TBs as inputs, outputs PBs of results along with intermediate data
  - Company conducted comparison tests among several I/O solutions. ADIOS is their choice for other codes: FWI, Kirchoff

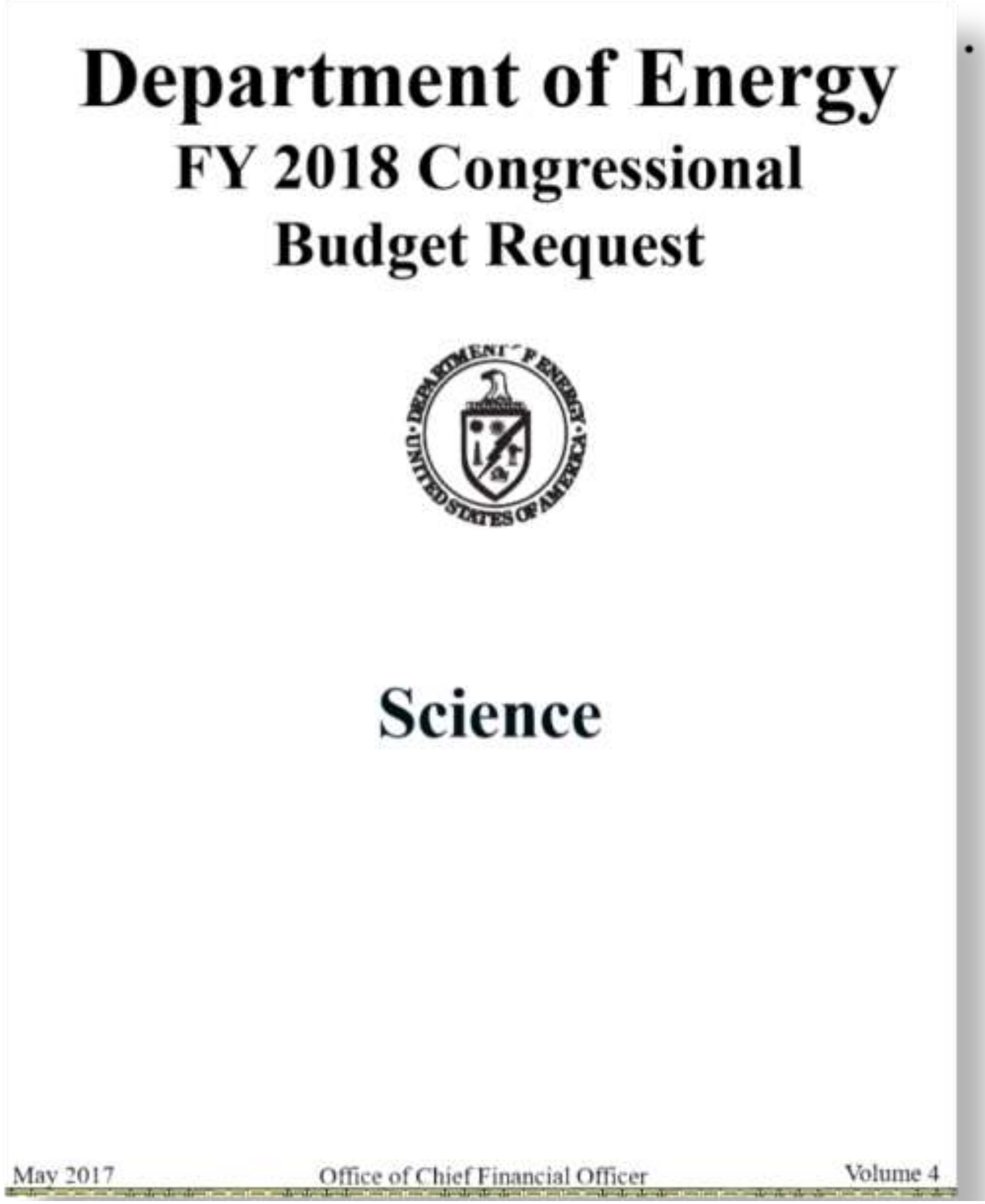
http://www.sciencedaily.com/releases/2016/01/160126130823.htm





Further impact: OpenFOAM CFD simulations

Yi Wang, Karl Meredith – FM Global
S. Klasky, N. Podhorszki - ORNL



Department of Energy
FY 2018 Congressional Budget Request

DEPARTMENT OF ENERGY UNITED STATES OF AMERICA

Science

May 2017 Office of Chief Financial Officer Volume 4

- Reducing the Damage Caused by Industrial Fires. Warehouse fires are the leading cause of commercial property damage, responsible for 40% of all industry property loss at a cost of approximately $188 million per year. Understanding how fires spread has the potential to save both business owners and insurance companies hundreds of millions of dollars. However, some of the most destructive fires – those that take place in mega-warehouses with ceilings up to 100 ft. high and a footprint in excess of 100,000 sq. ft.– are among the most difficult to study because they cannot be replicated in a test facility. To solve this problem, one of the world's largest commercial and industrial insurance companies partnered with the Oak Ridge Leadership Computing Facility to adapt an open-source fluid dynamics code to include the complex processes that occur during an industrial fire, including soot formation and sprinkler spray dynamics. After running their high resolution FireFOAM code on the Oak Ridge Leadership Computing Facility's Titan machine to learn how to stack storage boxes on pallets to impede the spread of horizontal flames, **the team incorporated the SciDAC-developed Adaptable I/O System (ADIOS) into FireFOAM to improve its efficiency in moving data on and off the supercomputer.** The new and improved code is now being used to simulate other commodities stored in warehouses, starting with large paper rolls. Both the results and the code are shared publicly to promote the improvement of fire protection standards across industry.

I/O in Seismic Tomography Workflow (PBs of data)

Scientific Achievement

Most detailed 3-D model of Earth's interior showing the entire globe from the surface to the core–mantle boundary, a depth of 1,800 miles.

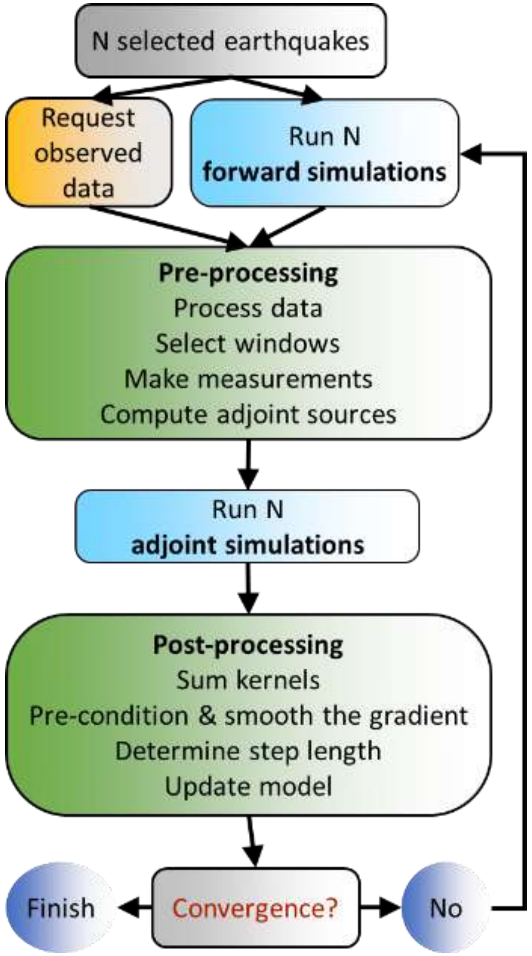
Significance and Impact

First global seismic model where no approximations were used to simulate how seismic waves travel through the Earth. Over 1 PB of data was generated in a 6 hour simulation

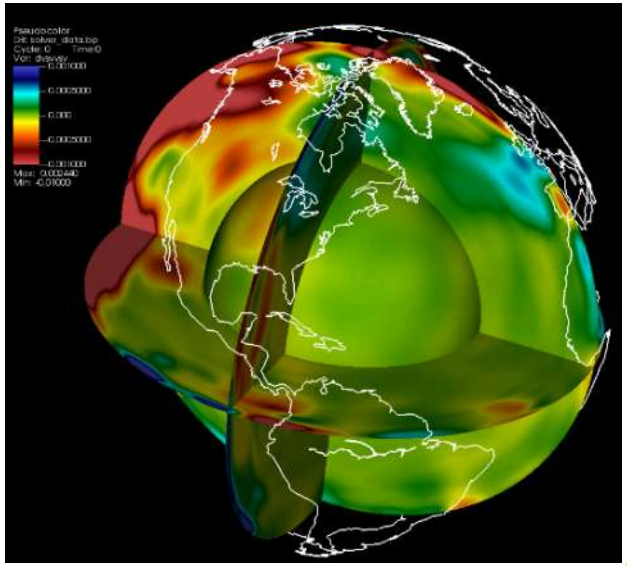
Research Details

- To improve data movement and flexibility, the Adaptable Seismic Data Format (ASDF) was developed that leverages the Adaptable I/O System (ADIOS) parallel library
- ASDF allows for recording, reproducing, and analyzing data on large-scale supercomputers
- 1PB of data is produced in a single workflow step, which is fully processed later in another step
- https://www.olcf.ornl.gov/2017/03/28/a-seismic-mapping-milestone

A

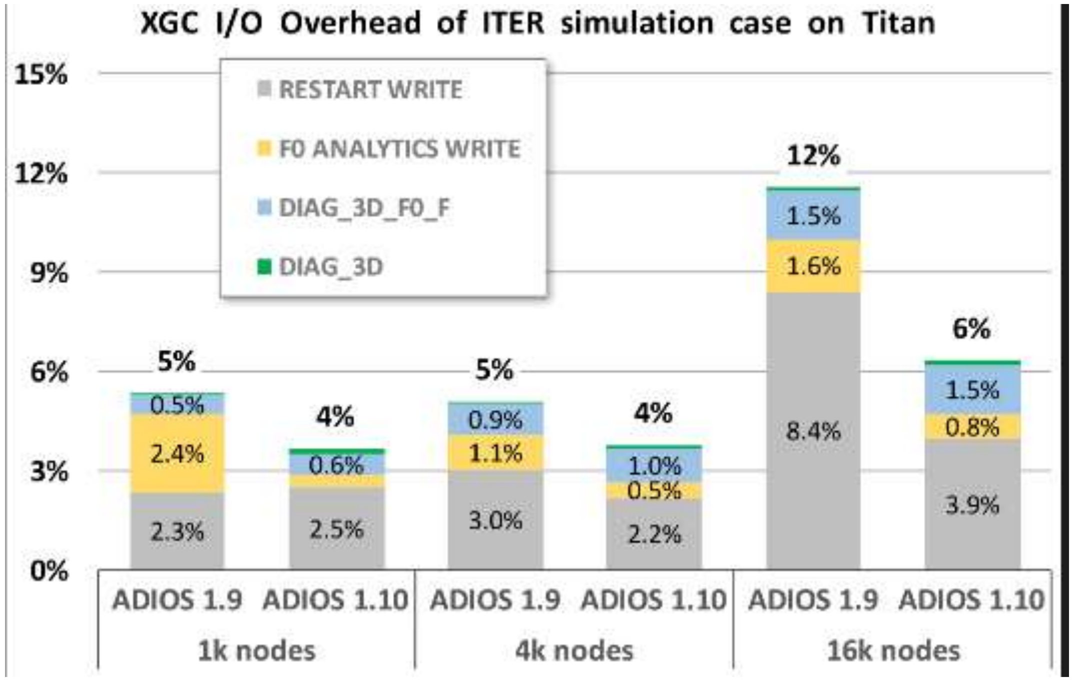


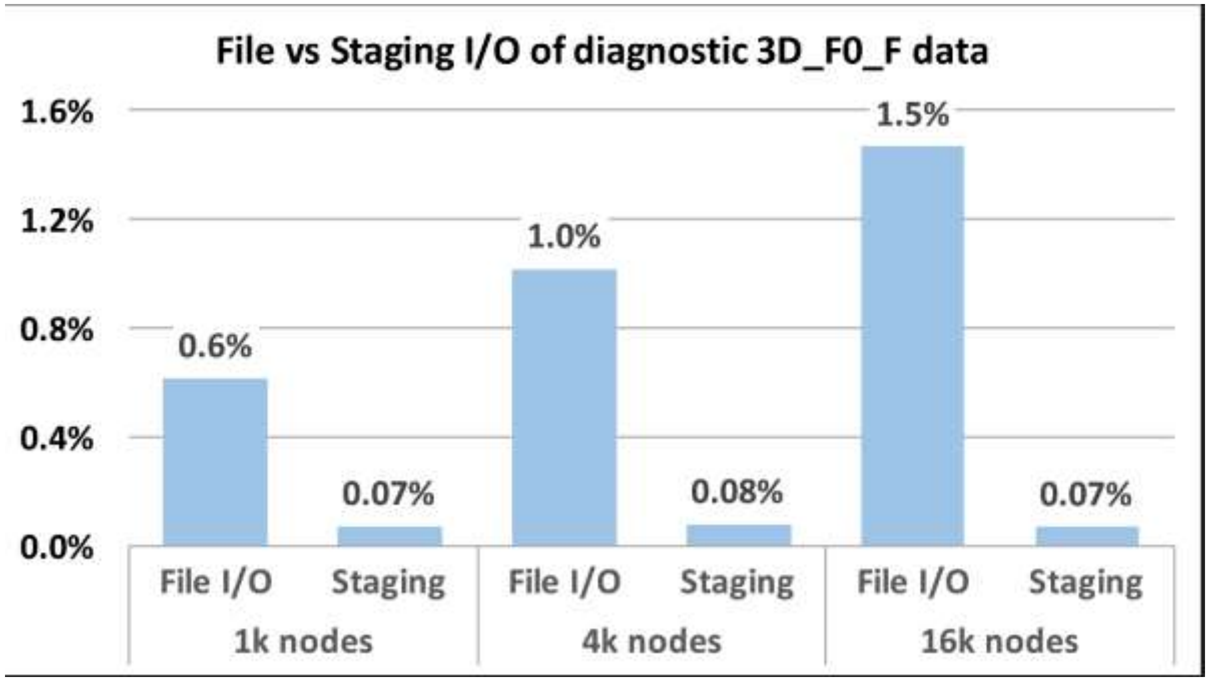
B



E. Bozdag; D. Peter; M. Lefebvre; D. Komatitsch; J. Tromp; J. Hill; N. Podhorszki; D. Pugmire. **Global adjoint tomography: first-generation model.** *Geophysical Journal International* 2016 207 (3): 1739-1766 https://doi.org/10.1093/gji/ggw356

Impact for Fusion Energy Science

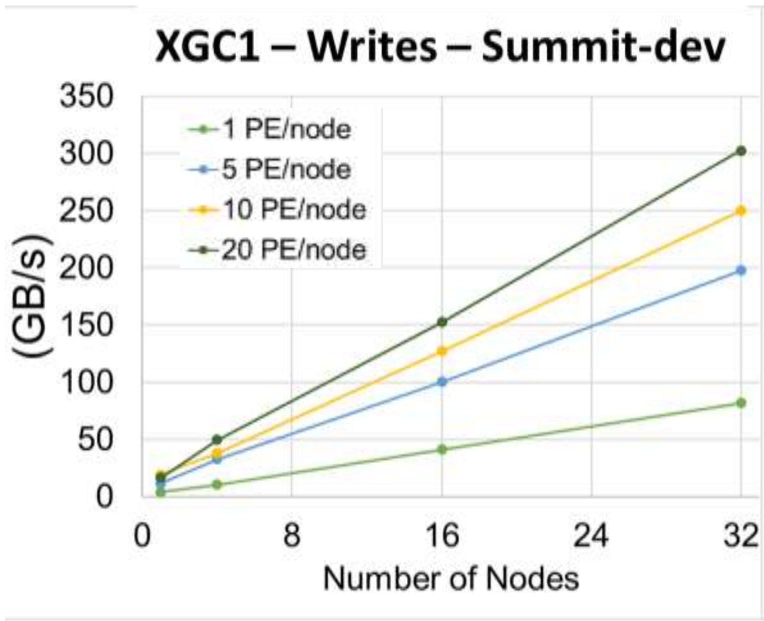


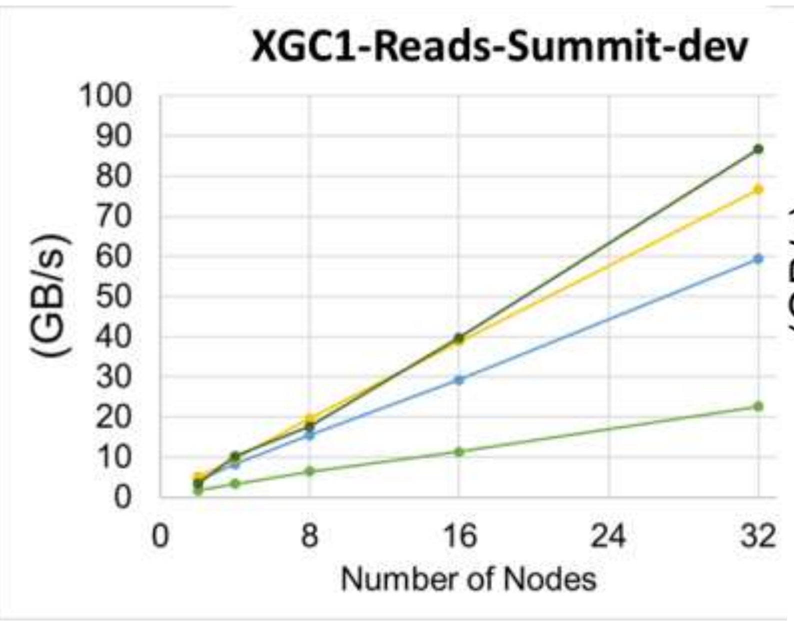


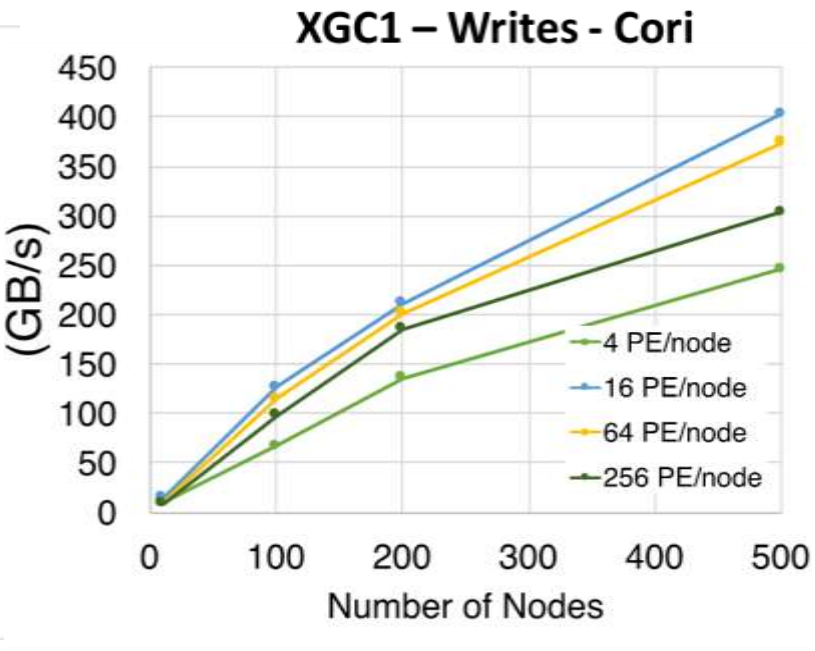
- 1 PB of total output per 24 hours, but really wanted 10 PB/24 hours

- J. Y. Choi, J. Logan, M. Wolf, G. Ostrouchov, T. Kurc, G. Liu, N. Podhorszki, S. Klasky, M. Romanus, Q. Sun, M. Parashar, R. M. Churchill, C.-S. Chang, TGE: Machine Learning Based Task Graph Embedding for Large-scale Topology Mapping in Cluster Computing (CLUSTER), 2017 IEEE International Conference on, IEEE.
- F. Zhang, T. Jin, Q. Sun, M. Romanus, H. Bui, S. Klasky, M. Parashar. In-memory staging and data-centric task placement for coupled scientific simulation workflows. Concurrency and Computation: Practice and Experience 2017, 29.
- J. Logan, J. Choi, M. Wolf, G. Ostrouchov, L. Wan, N. Podhorszki, W. Godoy, E. Lohrmann, G. Eisenhauer, C. Wood, K. Huck, S. Klasky, *Extending Skel to support the development and optimization of next generation I/O systems* in *Cluster Computing (CLUSTER), 2017 IEEE International Conference on*, IEEE.

ADIOS Burst Buffer performance for XGC1







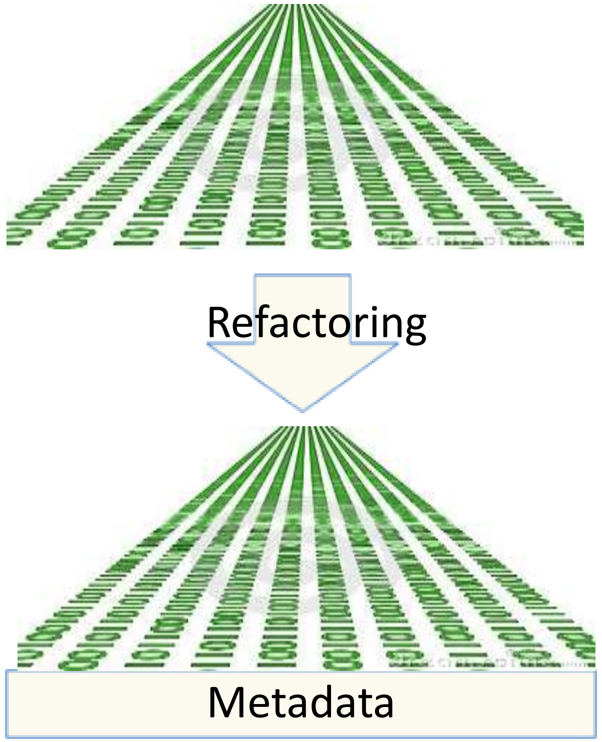
- 1 GB per PE written/read – in 20 PE/node case that means 20GB/node
- caveat: Plenty of free RAM available on each node for Summit-dev

Give them faster I/O and they write more

- ADIOS has accelerated the I/O of many communities by 10-100X over competing State of the Art Solutions
- Most of our users typically have a "time budget" for I/O
  - Increasing the I/O throughput → more data being output → moving from TBs to PB
  - For Experiments and Observations this means that we can push more data
- This created a new problem for the community
  - Too much data and nowhere to store this for the long term
- Data Refactoring (Reduction + Re-ordering)
  - "Bucket the data" into different levels of importance
  - i.e. Save the information in one bucket and the rest of the data in another
- **Goal is to reduce data sizes by 1,000X with minimal loss of information for later post processing**

Data Refactoring

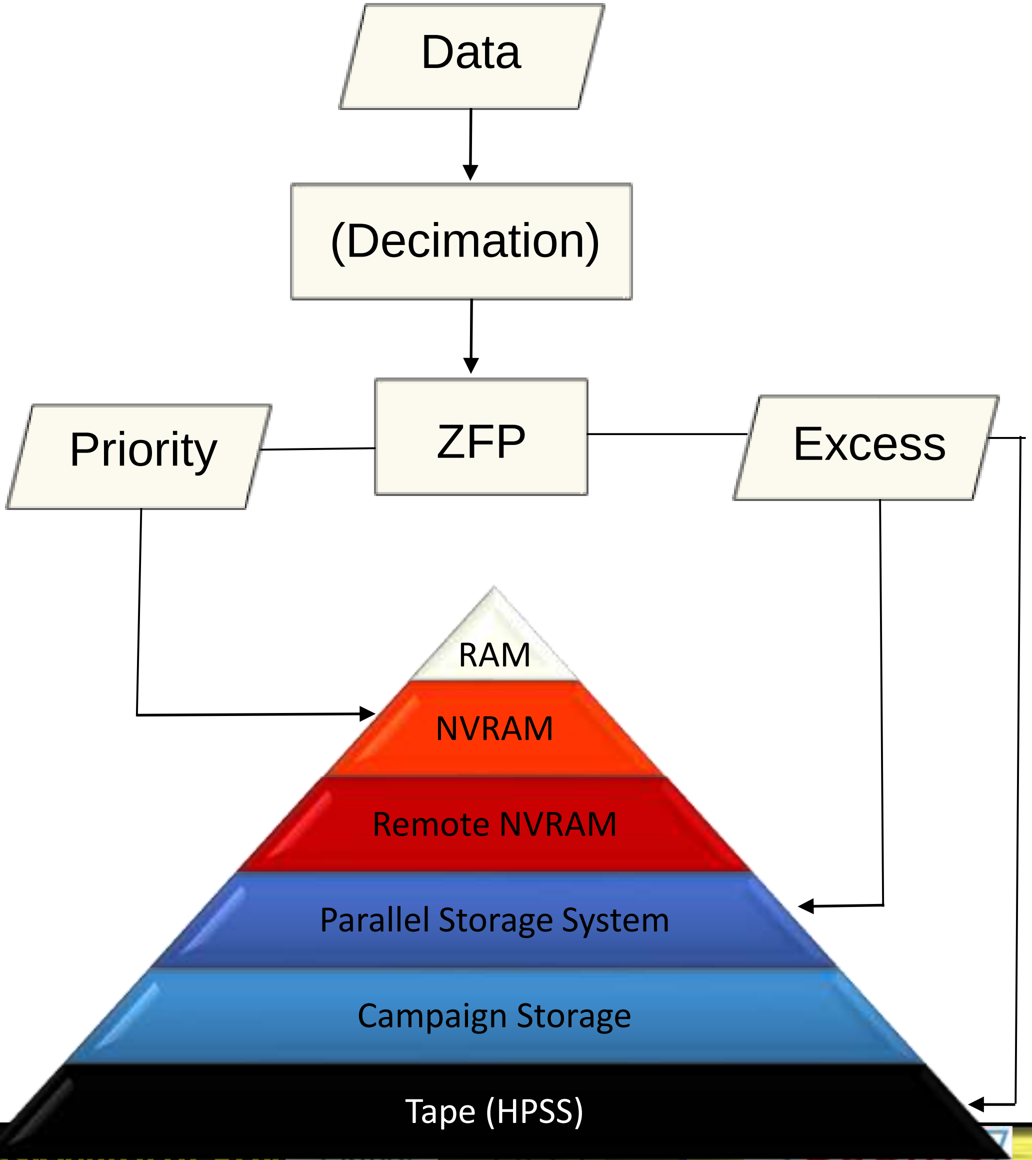
- Change internal data structure (schema) to make it easy to understand, modify, extend, and maintain, without affecting the external nature of the information.
- Challenges:
  - Identify and prioritize data access/usage scenarios
  - Choosing an appropriate representation
  - Testing external behavior
- Example:
  - Making data self-describing for fast read access:
    - Add headers with data bounds (min, max) and statistics (histogram)

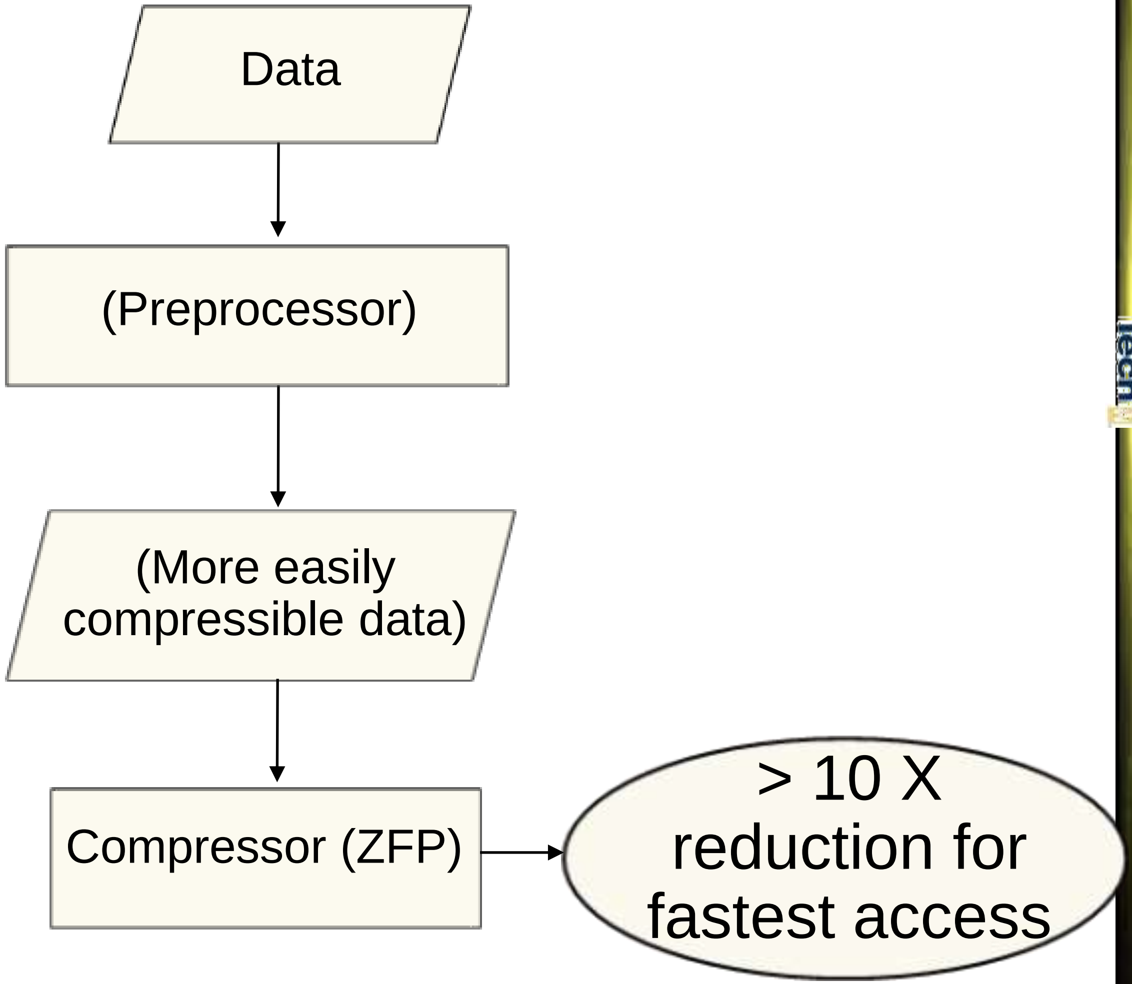


Field	Value
Size	100
Min	0
Max	1203213123
Dimensions	{2,50}
Type	float

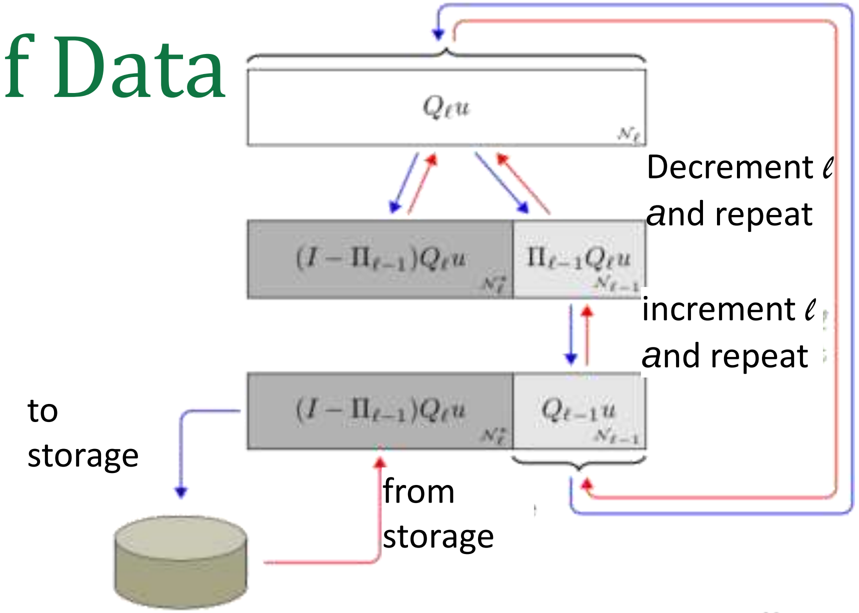
klasky@ornl.gov

We are exploring multiple methods to compress then utilize multi-tiered storage for speed vs. accuracy flexibility

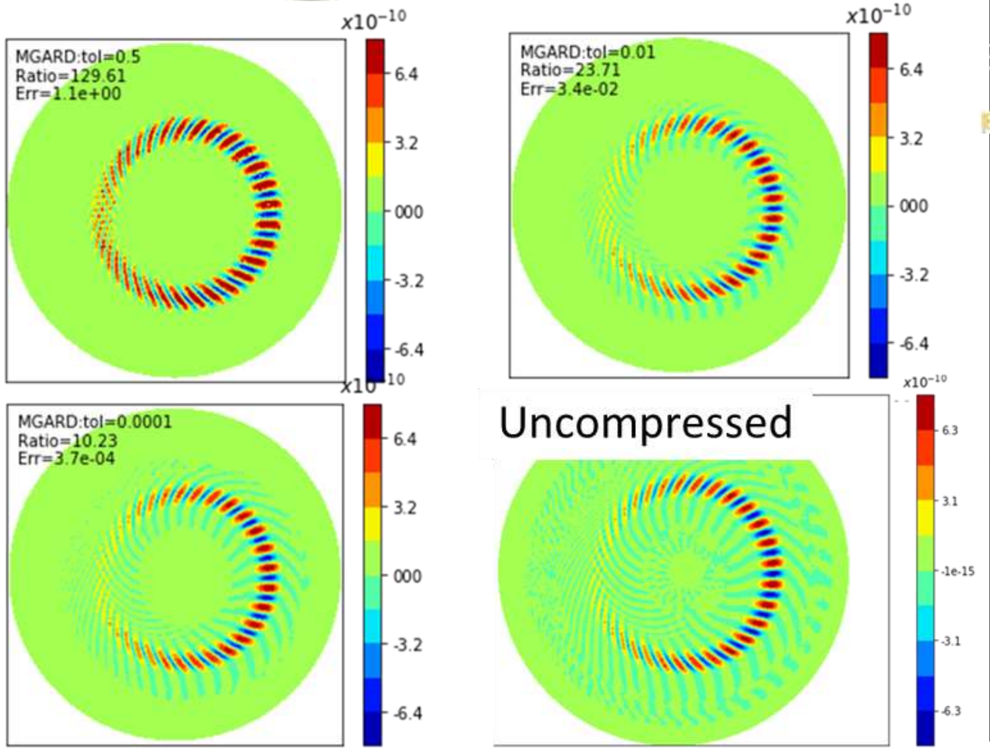




MGARD: MultiGrid Adaptive Reduction of Data



- Decomposes data into contributions from a hierarchy of meshes
- The hierarchical schema offers the flexibility to produce multiple levels of partial decompression of the data so users can work with reduced representations that require minimal storage while achieving the user specified tolerance
- Lossy data reduction based on discarding least important contributions
- Mathematically proven error bounds
- Applicable to structured (tensor product) grids with arbitrary spacing, integrated into ADIOS
- Aims to preserve structures present in input data



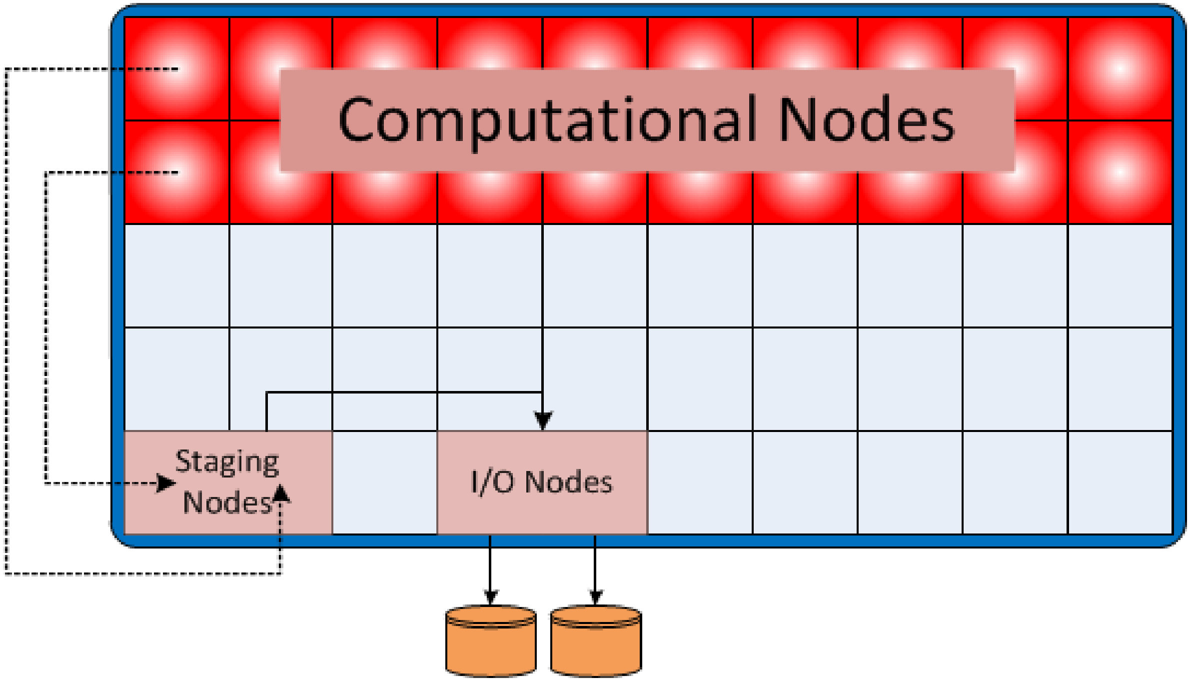
M. Ainsworth, O. Tugluk, B. Whitney, S. Klasky, "Multilevel Techniques for Compression and Reduction of Scientific Data --The Multivariate Case", SIAM Journal on Scientific Computing, Submitted for publication 2018.

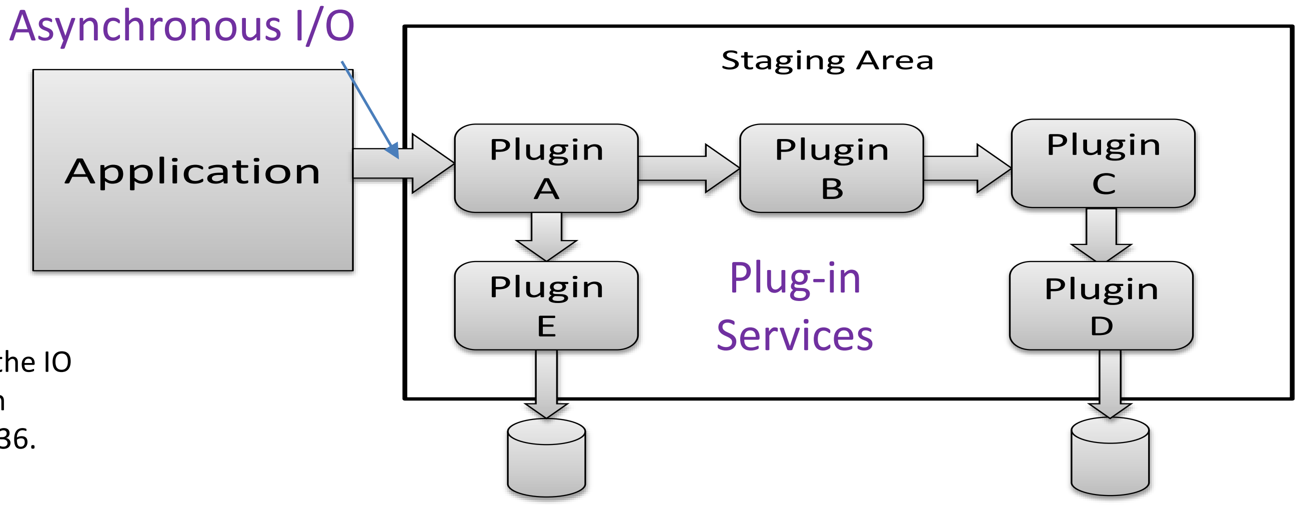
Making I/O more intelligent

- I/O tasks from HPC simulations are asynchronous
  - Only enforce synchronous behavior when/where necessary
  - Analysis, Visualization, diagnostics are a form of I/O tasks
  - Writing is an asymmetric task compared to reading
- We don't want to lose information for later post processing
- Need to store "more important" data on faster storage tiers
- Need to query data
- Do NOT treat data as a pile of bytes

Using Self Describing Data for Staging

- Goal: enhance data services and communication among applications providing an intermediate common area (staging) that reduces file system access costs.
- Self-describing data is crucial for making decisions on-the-fly at every “stage”.
- Imaging if this is done using only raw data?
- Components:
  - Asynchronous I/O buffers from Applications
  - Services provided as plugins:
    - Analytics & Visualization
    - Data Reduction
    - Data Transport (RDMA code coupling, files, WAN)



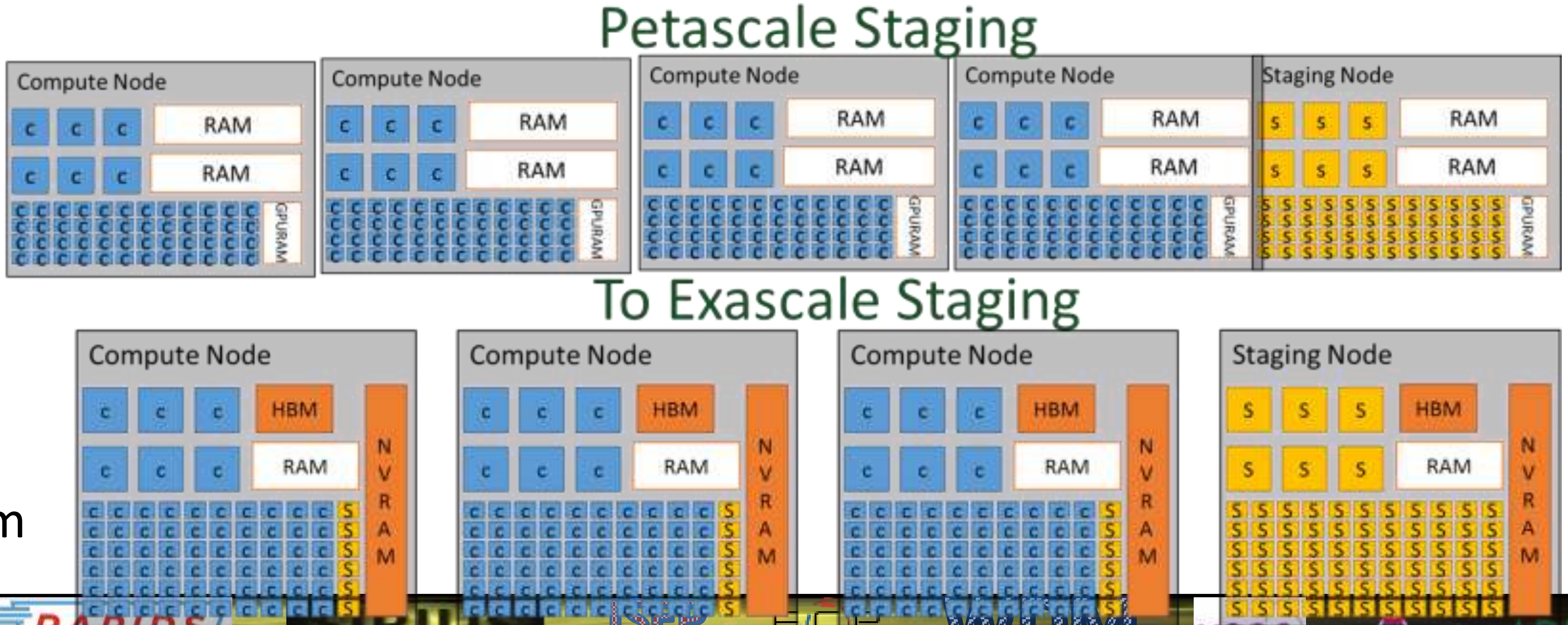


H. Abbasi, G. Eisenhauer, M. Wolf, K. Schwan, S. Klasky, Just in time: adding value to the IO pipelines of high performance applications with JITStaging in Proceedings of the 20th international symposium on High performance distributed computing, ACM, pp. 27–36.

klasky@ornl.gov

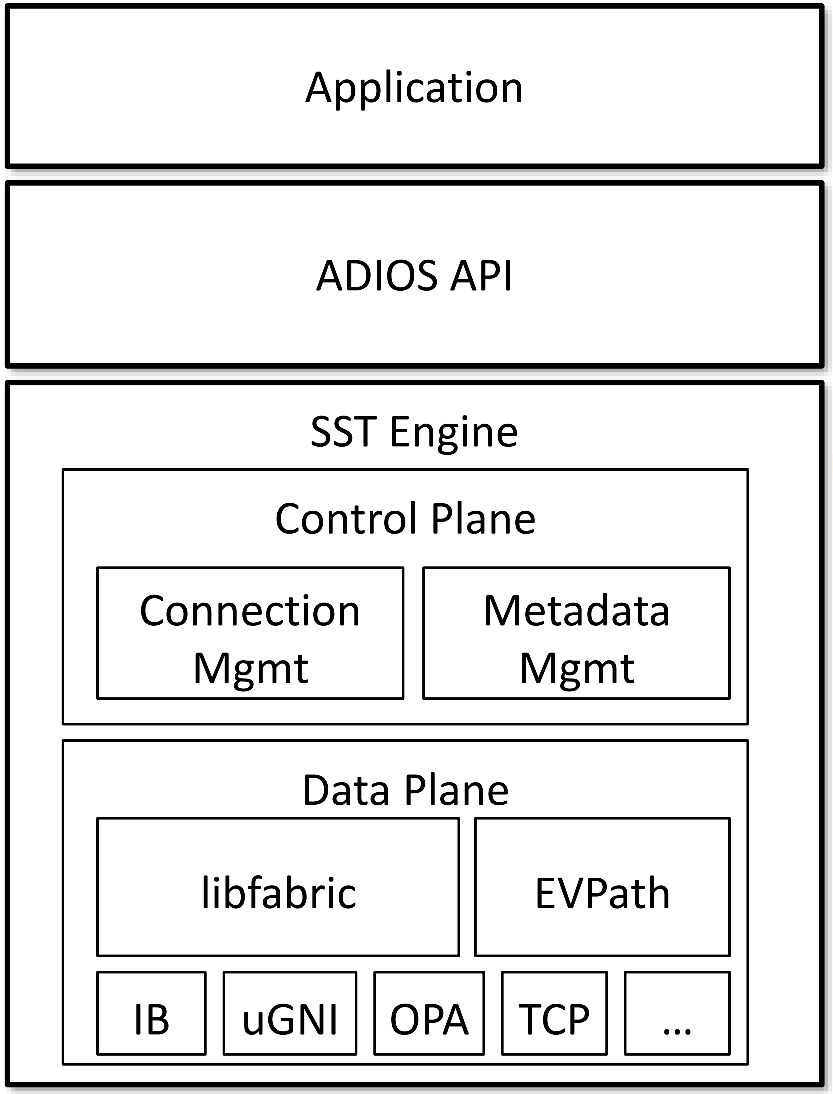
Staging

- Use compute and deep-memory hierarchies to optimize overall workflow for power vs. performance tradeoffs
- Abstract complex/deep memory hierarchy access
- Placement of analysis and visualization tasks in a complex system
- Impact of network data movement compared to memory movement
- Abstraction allows staging
  - On-same core
  - On different cores
  - On different nodes
  - On different machines
  - Through the storage system



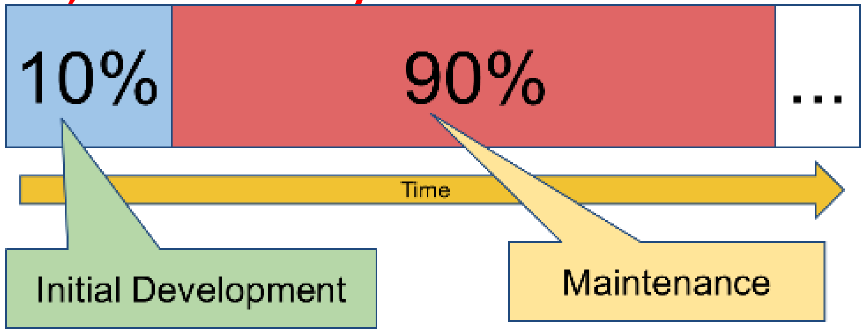
Sustainable Staging Transport (SST)

- Direct coupling between data producers and consumers for in-situ/in-transit processing
- Designed for portability and reliability.
- Control Plane
  - Manages meta-data and control using a message-oriented protocol
  - Inherits concepts from Flexpath, uses EVPath
  - Allows for dynamic connections, multiple readers and complex flow control management
- Data Plane
  - Exchange data using RDMA
  - Responsible for resource management for data transfer
  - Uses libfabric for portable RDMA support
  - Threaded to overlap communication with computation and for asynchronous progress monitoring
  - Modular interface with the control plane supports alternative data plane implementations



XSOA: eXtreme scale Service Oriented Architecture

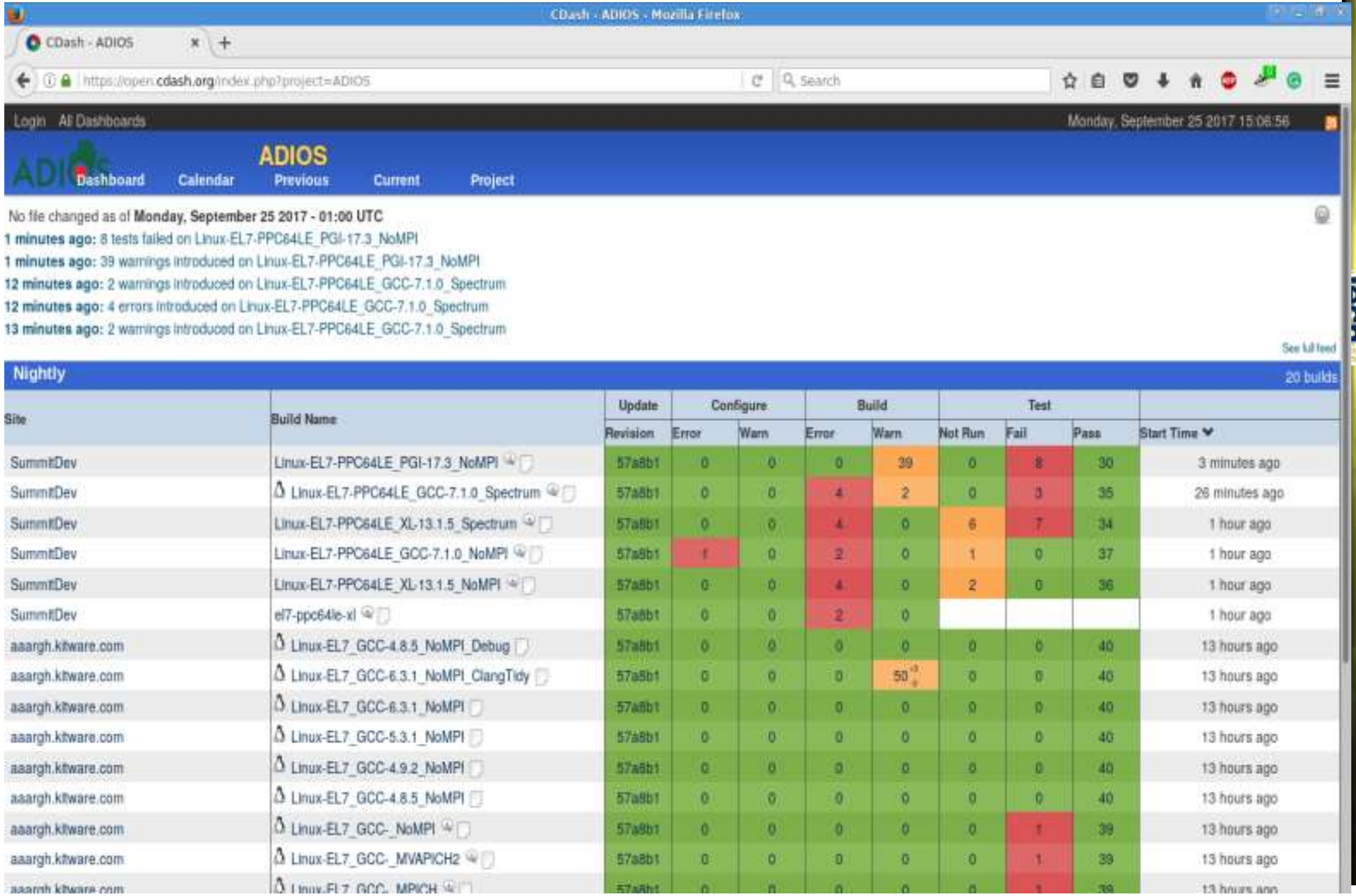
- Philosophy based on **Service-Oriented Architecture**
  - System management
  - Changing requirements
  - Evolving target platforms
  - Diverse, distributed teams
- Applications built by assembling services
  - Universal view of functionality
  - Well defined API
- Implementations can be easily modified and assembled
- Manage complexity while maintaining performance, scalability
  - Scientific problems and codes
  - Underlying disruptive infrastructure
  - Coordination across codes and research teams
  - End-to-end workflows



klasky@ornl.gov

Engineering ADIOS for Sustainability

- On-going effort to take what we've learned and build a better stack to support community engagement
- Re-engineering of ADIOS (ADIOS2) from the framework to the inside
  - Make the engagement at the tool/framework level as easy as possible.
  - Build the high performance core out to serve that.
- Uses community practices
  - Continuous integration
  - Github & C++
  - <u>Test-driven development based on applications</u>



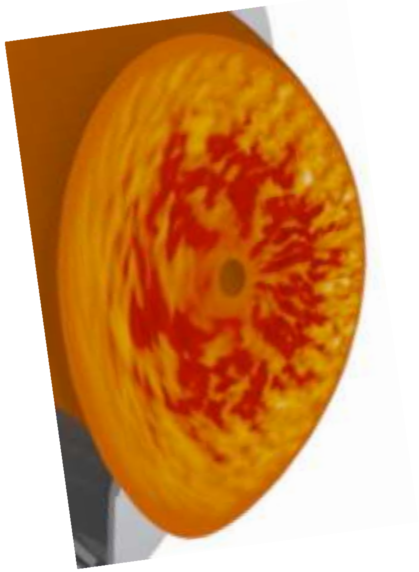
Site	Build Name	Update	Configure		Build		Test			
		Revision	Error	Warn	Error	Warn	Not Run	Fail	Pass	Start Time
SummitDev	Linux-EL7-PPC64LE_PGI-17.3_NoMPI	57a8b1	0	0	0	39	0	8	30	3 minutes ago
SummitDev	Linux-EL7-PPC64LE_GCC-7.1.0_Spectrum	57a8b1	0	0	4	2	0	3	35	26 minutes ago
SummitDev	Linux-EL7-PPC64LE_XL-13.1.5_Spectrum	57a8b1	0	0	4	0	6	7	34	1 hour ago
SummitDev	Linux-EL7-PPC64LE_GCC-7.1.0_NoMPI	57a8b1	1	0	2	0	1	0	37	1 hour ago
SummitDev	Linux-EL7-PPC64LE_XL-13.1.5_NoMPI	57a8b1	0	0	4	0	2	0	36	1 hour ago
SummitDev	el7-ppc64le-xl	57a8b1	0	0	2	0				1 hour ago
aaargh.kitware.com	Linux-EL7_GCC-4.8.5_NoMPI_Debug	57a8b1	0	0	0	0	0	0	40	13 hours ago
aaargh.kitware.com	Linux-EL7_GCC-6.3.1_NoMPI_ClangTidy	57a8b1	0	0	0	50	0	0	40	13 hours ago
aaargh.kitware.com	Linux-EL7_GCC-6.3.1_NoMPI	57a8b1	0	0	0	0	0	0	40	13 hours ago
aaargh.kitware.com	Linux-EL7_GCC-5.3.1_NoMPI	57a8b1	0	0	0	0	0	0	40	13 hours ago
aaargh.kitware.com	Linux-EL7_GCC-4.9.2_NoMPI	57a8b1	0	0	0	0	0	0	40	13 hours ago
aaargh.kitware.com	Linux-EL7_GCC-4.8.5_NoMPI	57a8b1	0	0	0	0	0	0	40	13 hours ago
aaargh.kitware.com	Linux-EL7_GCC-_NoMPI	57a8b1	0	0	0	0	0	1	39	13 hours ago
aaargh.kitware.com	Linux-EL7_GCC-_MVAPICH2	57a8b1	0	0	0	0	0	1	39	13 hours ago
aaargh.kitware.com	Linux-EL7_GCC-_MPICH	57a8b1	0	0	0	0	0	1	39	13 hours ago

GENE-core + XGC-edge coupling

- Core plasma is near thermal-equilibrium.
- Turbulence is ~1% perturbation: fast perturbative solution

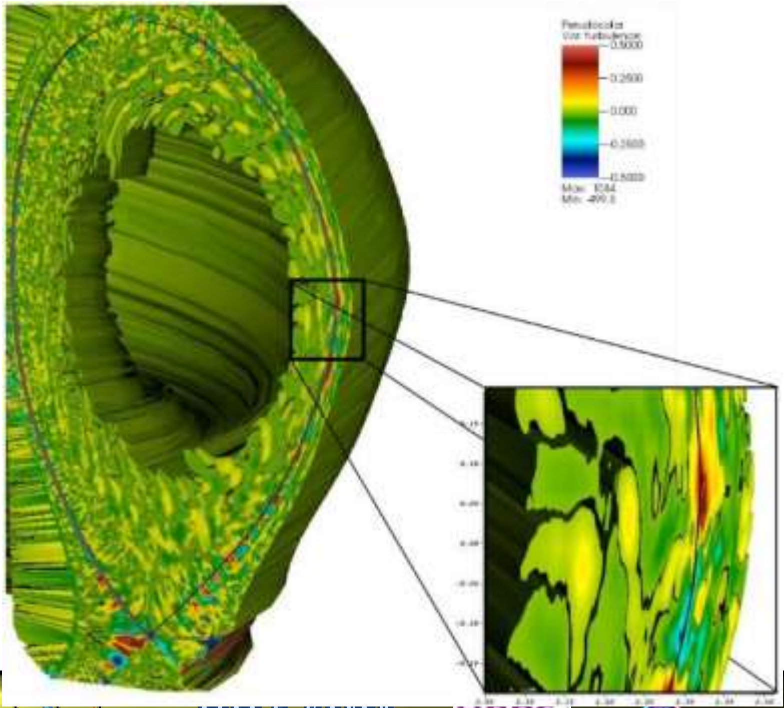
- Edge plasma is far-from equilibrium.
- Turbulence is scale-inseparable (≳10%).
- Neutral particle & atomic physics is important.
- ~Trillon particles for ITER, presently on Titan
  - Big data: exa bytes from Exascale computing

GENE-core



Can the turbulence patterns match up correctly at all time through a core-edge interface?

XGC-edge

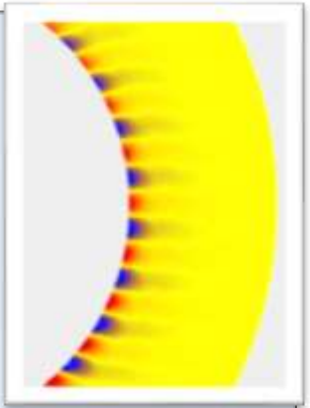


Steps to achieve our goals: 1: Provide high performance self describing I/O for each code, which can write to the burst buffer and the parallel file system, 1 API for C/R, for analysis, diagnostics

Code Coupling

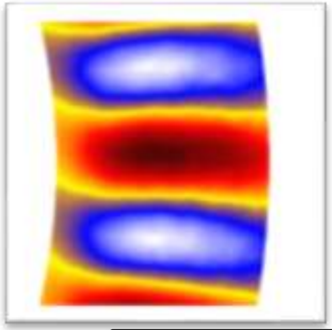
- Goal is to provide the capability to couple multiple codes through files, memory on the same node, memory on different nodes
- Codes are orchestrated from a "controller" service, which monitors, all of the different codes (executables)

XGC



- Initialize
- Begin Time loop
  - Evolve 5D particles
  - Charge Deposition
  - Read charge density from GENE
  - Poisson Solve
  - Write field to GENE
  - Output data to the storage system
- End time loop
- Finalize

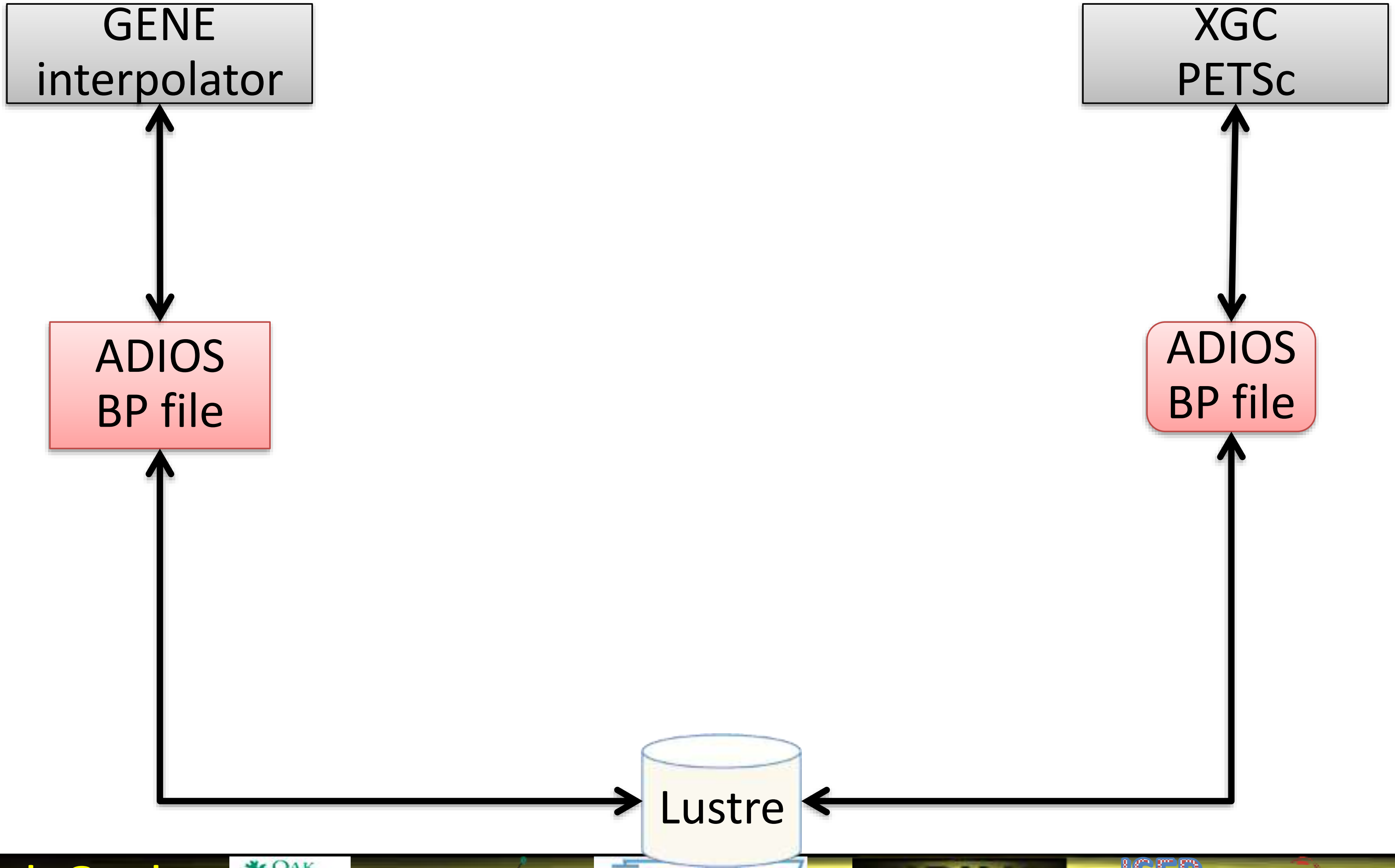
Overlapping region



GENE

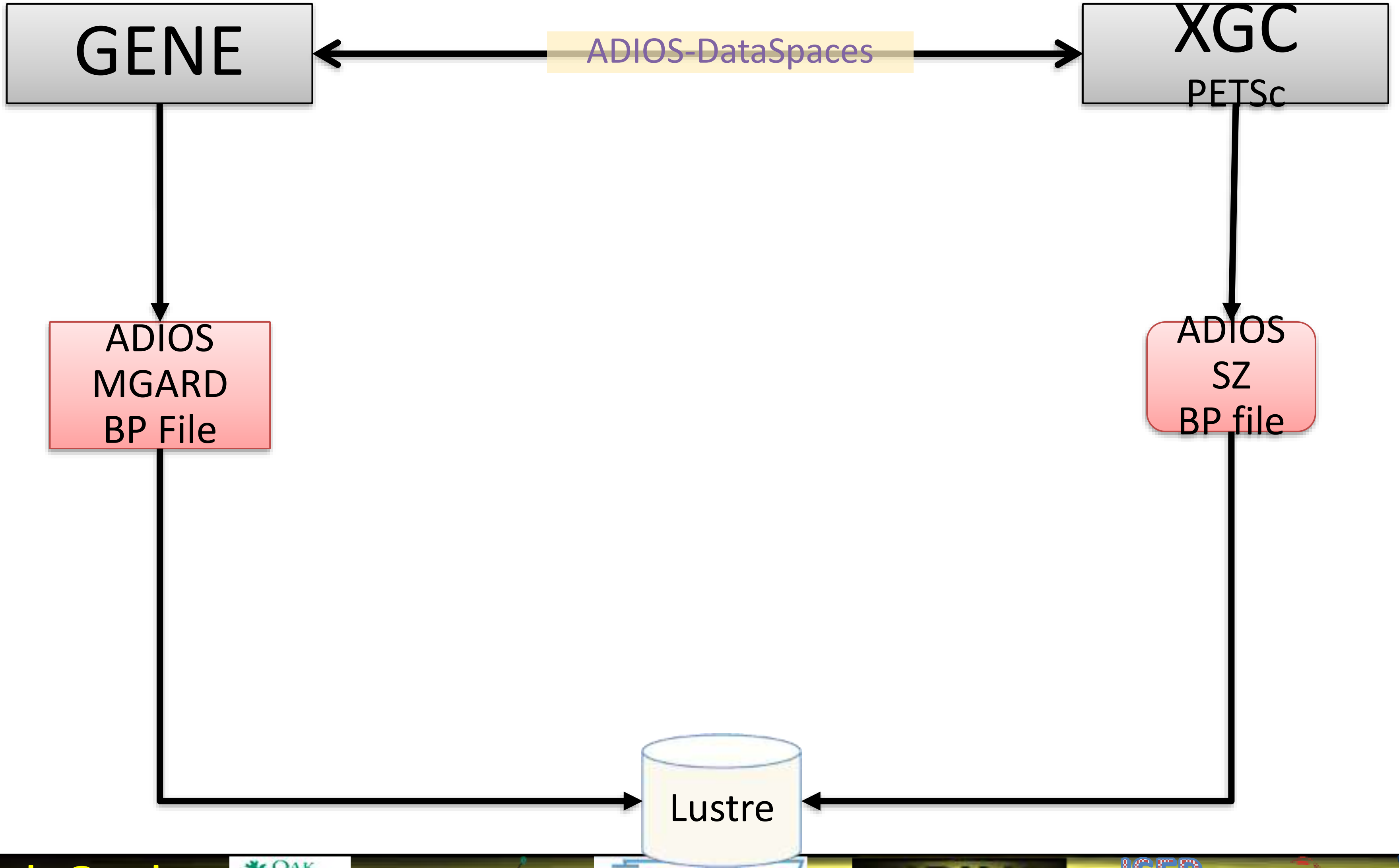
- Initialize
- Begin Time loop
  - Evolve 5D distribution function
  - Map charge density to XGC grid (interpolation)
  - Write charge density
  - Read field from XGC
  - Map field to GENE grid
  - Output data to the storage system
- End time loop
- Finalize

Steps to achieve our goals: 2: Couple the codes with files



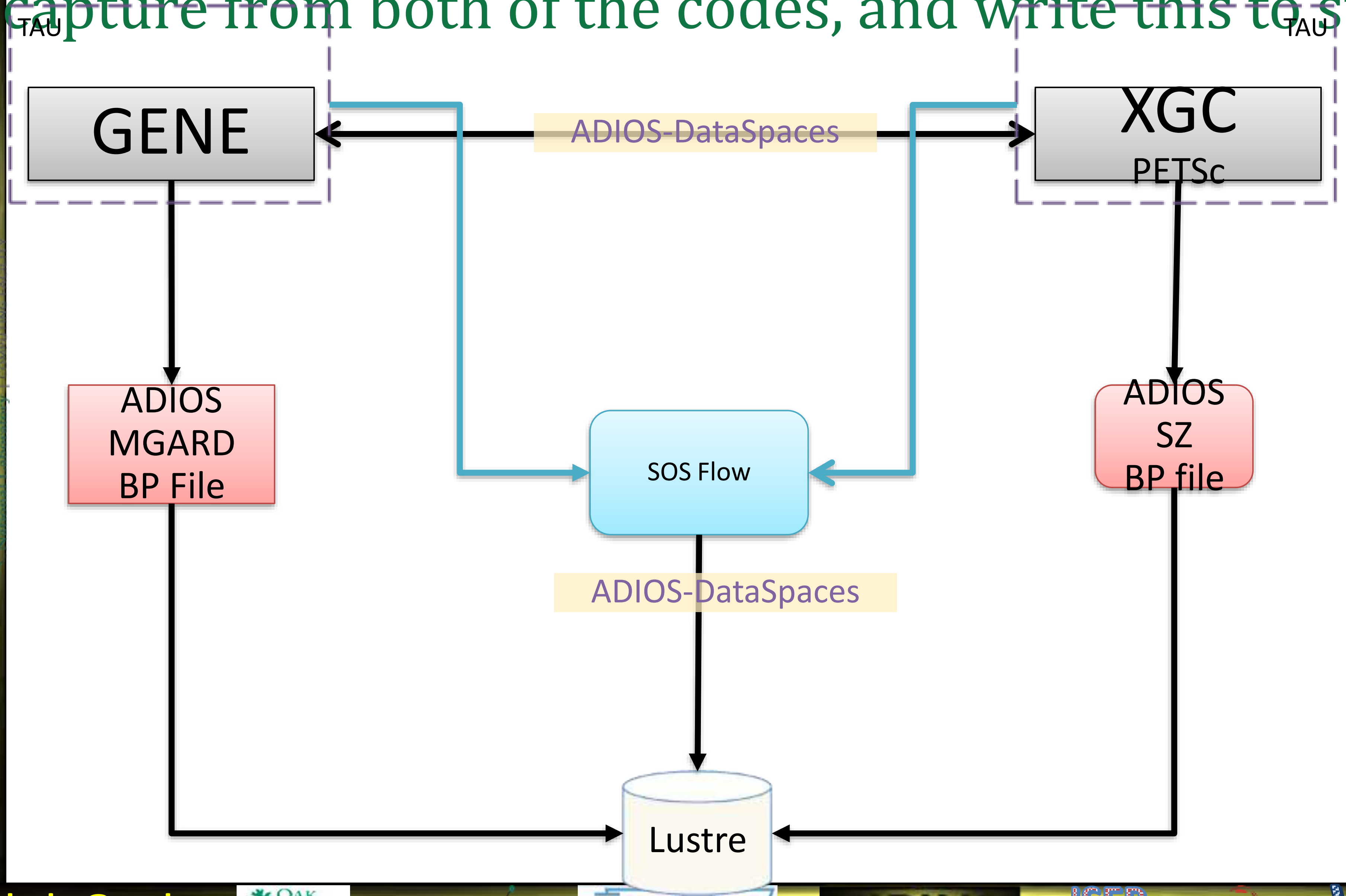
Change the code to be able to read in some of the variables from the other codes. Additionally add a routine to do the interpolation and turn off one of the field solvers in GENE

Steps to achieve our goals: 3: Couple the codes in memory, and reduce the data output to storage



No additional changes in the simulations

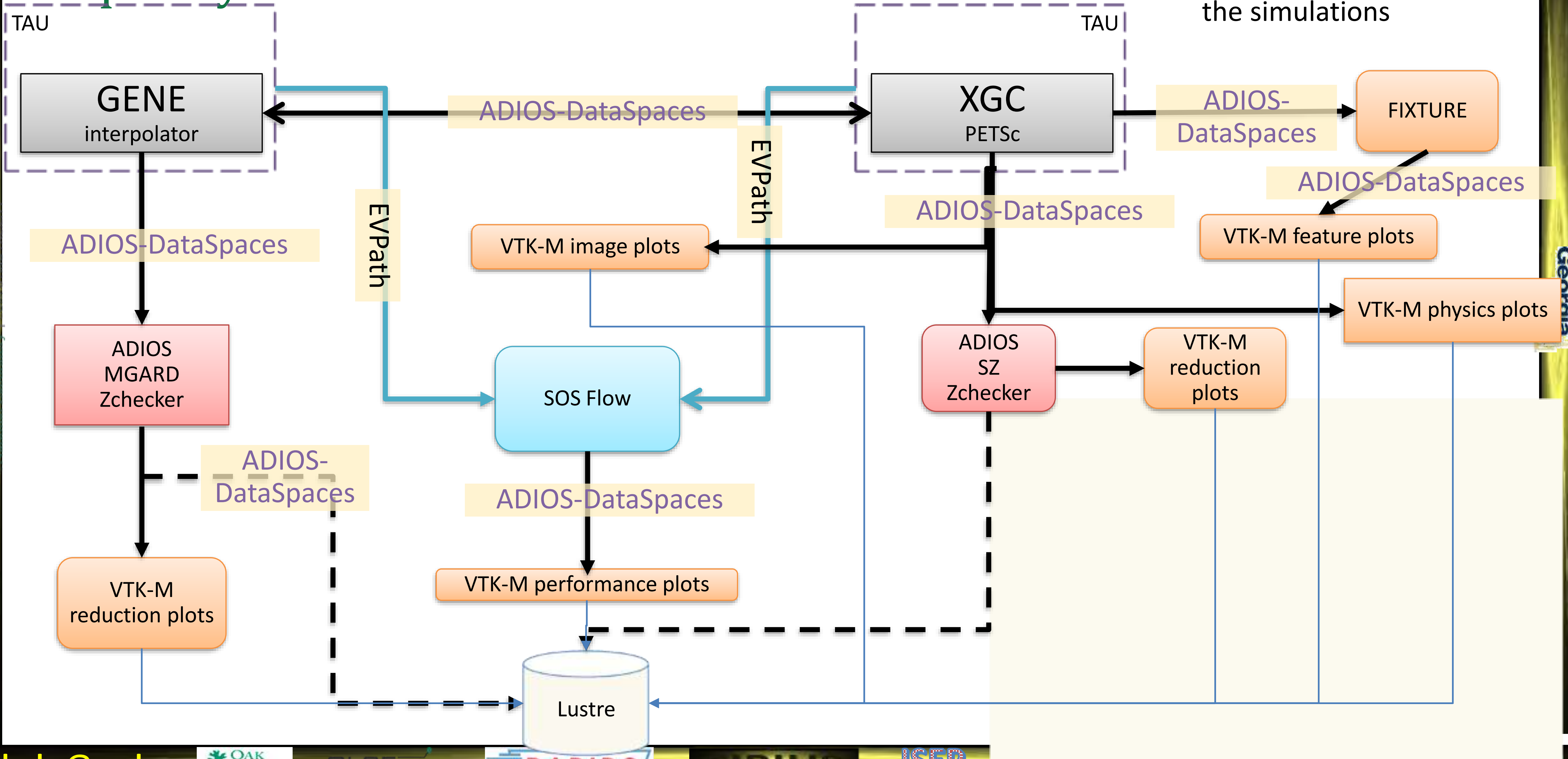
Steps to achieve our goals: 4: integrate real-time performance capture from both of the codes, and write this to storage



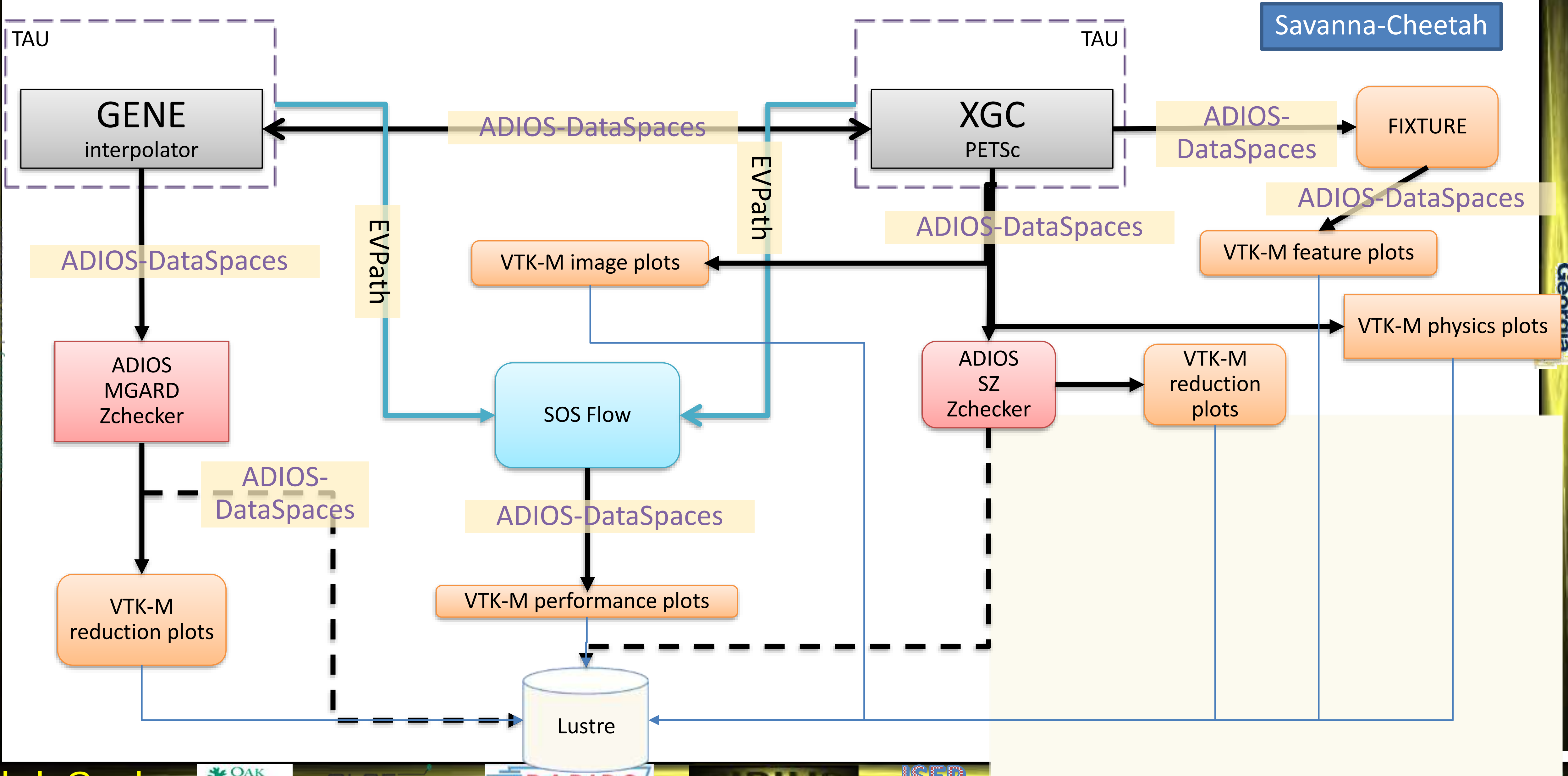
No additional changes in the simulations

Steps to achieve our goals: 5: Add visualization services and check the quality of the reduction

No additional changes in the simulations



Steps to achieve our goals: 6: Examine the Co-design tradeoffs



Desktop 1 for the demo

XGC1 Output

```
XGC
 - turb solver memory create
 create_solver: first vertex in domain: F
CREATE_SOLVER: make partitioning with  207312/ 263243 real vertices
 prepare gyro-averaging matrix
 ********** solve_poisson_private static variable initialized ************
 setup poloidal decomposition
 initial diagnosis
 dump_bfield
 background elec charge
 End of setup
 main loop started
partition_opt: (using Alg. 2)  0.26122E-02
partition_opt: (alt. Alg. 1)   0.26245E-02   0.29983E-06     10
 set_weights: increased ion    maxnum from,to:        12000        13176
 - do loop start
step,trigger,ratio,# of ion      1    1.1979     1.0794      30720000
step,f0(ratio,max)    1    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step,trigger,ratio,# of ion      2    1.1979     1.0790      30720000
step,f0(ratio,max)    2    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step,trigger,ratio,# of ion      3    1.1979     1.0761      30720000
step,f0(ratio,max)    3    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step,trigger,ratio,# of ion      4    1.1979     1.0762      30720000
step,f0(ratio,max)    4    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step,trigger,ratio,# of ion      5    1.1979     1.0736      30720000
step,f0(ratio,max)    5    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step,trigger,ratio,# of ion      6    1.1979     1.0713      30720000
step,f0(ratio,max)    6    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
```

GENE Output

```
GENE
 wait from XGC (stage, time,itime)     [illegible]   2.0058764154965034E-002
 send to XGC(stage, time,itime)        [illegible]
 wait from XGC (stage, time,itime)        3   2.0058764154965034E-002
 send to XGC(stage, time,itime)        4   2.0058764154965034E-002
 wait from XGC (stage, time,itime)        4   2.0058764154965034E-002
 send to XGC(stage, time,itime)        1   3.0088146232447550E-002
 wait from XGC (stage, time,itime)        1   3.0088146232447550E-002
   0.030088
  1.0830E-21  6.0385E-25  1.1277E-24  6.9881E-23  1.4328E-22  0.0000E+00  1
 send to XGC(stage, time,itime)        2   3.0088146232447550E-002
 wait from XGC (stage, time,itime)        2   3.0088146232447550E-002
 send to XGC(stage, time,itime)        3   3.0088146232447550E-002
 wait from XGC (stage, time,itime)        3   3.0088146232447550E-002
 send to XGC(stage, time,itime)        4   3.0088146232447550E-002
 wait from XGC (stage, time,itime)        4   3.0088146232447550E-002
 send to XGC(stage, time,itime)        1   4.0117528309930067E-002
 wait from XGC (stage, time,itime)        1   4.0117528309930067E-002
   0.040118
  1.0808E-21  1.0487E-24  1.9593E-24  6.9317E-23  1.4301E-22  0.0000E+00  2
 send to XGC(stage, time,itime)        2   4.0117528309930067E-002
 wait from XGC (stage, time,itime)        2   4.0117528309930067E-002
 send to XGC(stage, time,itime)        3   4.0117528309930067E-002
 wait from XGC (stage, time,itime)        3   4.0117528309930067E-002
 send to XGC(stage, time,itime)        4   4.0117528309930067E-002
 wait from XGC (stage, time,itime)        4   4.0117528309930067E-002
 send to XGC(stage, time,itime)        1   5.0146910387412584E-002
 wait from XGC (stage, time,itime)        1   5.0146910387412584E-002
   0.050147
  1.0784E-21  1.5987E-24  3.0223E-24  6.8843E-23  1.4279E-22  0.0000E+00  2
 send to XGC(stage, time,itime)        2   5.0146910387412584E-002
 wait from XGC (stage, time,itime)        2   5.0146910387412584E-002
 send to XGC(stage, time,itime)        3   5.0146910387412584E-002
 wait from XGC (stage, time,itime)        3   5.0146910387412584E-002
```

Z-checker output XGC1

```
XGC_SZ_compression_and_Z-Check
    double        dpot[263243, 8] :
Rank 0: allocate 16.0672 MB for input buffer
[ZC] Reading ZC configuration file (zc.config) ...

File info:
  current step:    4
  last step:       4
  # of variables: 4:
Get info on variable 0: nnode
    integer     nnode     scalar
Get info on variable 1: nphi
    integer     nphi      scalar
Get info on variable 2: iphi
    integer     iphi      scalar
Get info on variable 3: dpot
    double      dpot[263243, 8] :
Rank 0: allocate 16.0672 MB for input buffer
[ZC] Reading ZC configuration file (zc.config) ...

File info:
  current step:    5
  last step:       5
  # of variables: 4:
Get info on variable 0: nnode
    integer     nnode     scalar
Get info on variable 1: nphi
    integer     nphi      scalar
Get info on variable 2: iphi
    integer     iphi      scalar
Get info on variable 3: dpot
    double      dpot[263243, 8] :
Rank 0: allocate 16.0672 MB for input buffer
[ZC] Reading ZC configuration file (zc.config) ...
```

XGC1 Performance

```
./XGC/timing/timing_000000005.txt.0000
                              On   Called  Recurse    Wallclock
 MAIN_LOOP                    -      5        -      123.386215
   IPC_LOOP                   -     20        -      123.375244
     CHARGEI                  -     20        -       65.803413
       CHARGEI_SEARCH_INDEX   -     20        -        0.362295
       CHARGEI_SCATTER        -     20        -        3.691921
       CHARGEI_GYRO_AVG       -     20        -        7.683217
       CHARGEI_GA_RED_BCAST   -     20        -        0.407333
     POISSON                  -     20        -       18.490120
       POISSON_00MODE         -     20        -        9.138186
       POISSON_TURB           -     20        -        8.058781
       POISSON_SR_POT         -     20        -        1.244160
     CCE_SEND_FIELD           -     20        -        0.154491
     CCE_RECEIVE_FIELD        -     20        -        0.012654
     CCE_PROCESS_FIELD        -     20        -        0.000021
     GET_POT_GRAD             -     20        -       32.885841
       GET_POT_LOOPS          -     20        -        2.098544
       GET_POT_CNVRT          -     40        -        2.718233
       GET_POT_SR             -     20        -        0.668561
       GET_POT_MAT_MULT       -     20        -        1.999560
       GET_POT_BCAST          -     20        -       12.111038
       GET_POT_IDX_REORD      -     20        -       12.168242
     PUSH_I                   -     20        -        0.270981
       PUSH_LOOP              -     20        -        0.268854
     DIAGNOSIS                -     20        -        0.730907
       DIAG_1D_OUTPUT         -      5        -        0.669580
       DIAG_3D                -      5        -        0.059799
     SHIFT_I                  -     20        -        2.293785
       MEM_CLEAN_I            -     20        -        0.002157
   COLLISION                  -      5        -        0.000045

head sum =  0.000291 wallclock seconds
```

Savanna

```
VTKM
  CSE: 3
  CSE: 4
  CSE: 5
  CSE: 6
  CSE: 7
Streamer Energy = 1.11266e-13
Total Energy    = 1.13239e-13
Compute Streamer Energy :: Wall Time = 2162.85 ms

CSA: 0
CSA: 1
CSA: 2
CSA: 3
CSA: 4
CSA: 5
CSA: 6
CSA: 7
Compute Streamer Area :: Wall Time = 49.0717 ms

Compute Growth :: Wall Time = 5.73875 ms

Threshold: -100 0.276
Threshold: 0.467 100
Threshold: 0.276 0.467
Threshold: -3.2375e-10 -3.2375e-11
Threshold: 3.2653e-11 3.2653e-10
3D Views :: Wall Time = 8363.98 ms

Compute Ballooning :: Wall Time = 3367.18 ms

  CSE: 0
  CSE: 1
  CSE: 2
```

Z-checker output GENE

```
GENE_MGARD_compression_and_Z-Check

File info:
  current step:    5
  last step:       5
  # of variables: 10:
Get info on variable 0: nx
    integer     nx        scalar
Get info on variable 1: nz
    integer     nz        scalar
Get info on variable 2: gx
    integer     gx        scalar
Get info on variable 3: gz
    integer     gz        scalar
Get info on variable 4: ox
    integer     ox        scalar
Get info on variable 5: oz
    integer     oz        scalar
Get info on variable 6: time
    double      time      scalar
Get info on variable 7: timestep
    integer     timestep  scalar
Get info on variable 8: phi_real
    double      phi_real[24, 160] :
Get info on variable 9: phi_imag
    double      phi_imag[24, 160] :
Rank 0: allocate 0.0294189 MB for input buffer
[ZC] Reading ZC configuration file (zc.config) ...
Got: 4    2e-05
Tog: 17          129
[ZC] Reading ZC configuration file (zc.config) ...
Got: 4    2e-05
Tog: 17          129
```

```
XGC
 (t_initf)       profile_single_file=      F
 (t_initf)       profile_global_stats=     T
 (t_initf)       profile_ovhd_measurement= F
 (t_initf)       profile_add_detail=       F
 (t_initf)       profile_papi_enable=      F
 call petsc_init
 staging_read_method_name=DIMES
 staging_read_method=            4
 # of OMP threads =             1
 Total simulation processors :          3072
 setting up.........            0
 Read input file
 sml_param
 ptl_param
 eq_param
 col_param
 diag_param
 smooth_param
 mon_param
 Interplane-Major ordering for sml_comm communicator
 plane mpi communication
 inter-plane mpi communication
 adios mpi communication?
 end of commnunicator setup
 reading......
 eq end_flag           -1
 ptl_num for each CPU =       10000
 memory allocation
 init_interpolation
 third difinition part
 eq_x_slope=   0.0000000000000000
 eq_x2_slope=  -0.0000000000000000
```

```
GENE
   z-proc:          1.571             2.094             1.581
   z-proc:         -1.571            -1.047            -1.570          -0
 blocks           1      15941       53601
   x-proc.      3       0.7002             0.8953             0.6948
   z-proc.  3    28     88    115
   z-proc:         -2.356            -1.833            -2.355          -
   x-proc.      1       0.3000             0.4951             0.2984
   x-proc.      0       0.9996E-01         0.2950             0.9951E-01
   z-proc:        -0.7854           -0.2618           -0.7540          -0
   z-proc.  0    29      0     28
   z-proc:         -3.142            -2.618            -3.142          -
   z-proc:         0.7854             1.309            0.8004
   z-proc:          2.356             2.880             2.369
   z-proc.  4    29    116    144
   z-proc:          0.000            0.5236            0.4261E-02        0
   x-proc.      2    40     80   119   65598
 blocks           2      53602      119199
   x-proc.      2       0.5001             0.6952             0.4970
Time for matrix inversion:     0.057 sec
Time for initializing poloidal planes:     0.839 sec

initialization: XGC
  For XGC comparison, amplitude 5.000E-11
  gaussian in x, sigma_i= 2.700E+01

linear computation
4th order Runge-Kutta
Initializing the FLR Correction prefactors for global calculation.

*** entering time loop ***
maximal        4000 timesteps.
 send to XGC(stage, time,itime)           1   0.0000000000000000
 wait from XGC (stage, time,itime)          1   0.0000000000000000
```

```
XGC_SZ_compression_and_Z-Check
Input stream              = xgc.dpot.edge.bp
Output stream             = SZ-compressed.bp
Read method               = DIMES (id=4)
Read method parameters    = "verbose=1"
Write method              = DIMES
Write method parameters   = "verbose=1"
Variable to transform     = "dpot"
Transform parameters      = "SZ:rel=0.001,zcheck"

Waiting to open stream xgc.dpot.edge.bp...
```

```
No performance data yet.
```

```
VTKM

        Running ./showcaseVisualizations with:
                --quad-zoffset=0.15
                --mesh-file=xgc.mesh.bp
                --field-file=xgc.3d.edge.bp
                --read-method=dimes
                --time-start=0
                --time-end=4000
```

```
GENE_MGARD_compression_and_Z-Check
Input stream              = outfiles/field-ky0.dat.bp
Output stream             = field-ky0-out-mgard.bp
Read method               = DIMES (id=4)
Read method parameters    = "verbose=1"
Write method              = DIMES
Write method parameters   = "verbose=1"
Variable to transform     = "phi_real,phi_imag"
Transform parameters      = "MGARD:tol=0.0001,zcheck"

Waiting to open stream outfiles/field-ky0.dat.bp...
```

bash

```
XGC
step.trigger.ratio.# of ion      11     1.1979     1.0759        30720000
step.f0(ratio.max)    11    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      12     1.1979     1.0770        30720000
step.f0(ratio.max)    12    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      13     1.1979     1.0711        30720000
step.f0(ratio.max)    13    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      14     1.1979     1.0753        30720000
step.f0(ratio.max)    14    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      15     1.1979     1.0826        30720000
step.f0(ratio.max)    15    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      16     1.1979     1.0837        30720000
step.f0(ratio.max)    16    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      17     1.1979     1.0743        30720000
step.f0(ratio.max)    17    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      18     1.1979     1.0716        30720000
step.f0(ratio.max)    18    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      19     1.1979     1.0776        30720000
step.f0(ratio.max)    19    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      20     1.1979     1.0735        30720000
step.f0(ratio.max)    20    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
step.trigger.ratio.# of ion      21     1.1979     1.0716        30720000
step.f0(ratio.max)    21    -1.0000     0.0000
 2d particle and heat flux is not ready yet for deltaf-f method
```

bash

```
GENE
 wait from XGC (stage, time,itime)          2  0.17049949531720285
 send to XGC(stage, time,itime)           3  0.17049949531720285
 wait from XGC (stage, time,itime)          3  0.17049949531720285
 send to XGC(stage, time,itime)           4  0.17049949531720285
 wait from XGC (stage, time,itime)          4  0.17049949531720285
 send to XGC(stage, time,itime)           1  0.18052887739468537
 wait from XGC (stage, time,itime)          1  0.18052887739468537
    0.180529
  1.0419E-21  1.4312E-23  3.6619E-23  7.0670E-23  1.3914E-22  0.0000E+00  2
 send to XGC(stage, time,itime)           2  0.18052887739468537
 wait from XGC (stage, time,itime)          2  0.18052887739468537
 send to XGC(stage, time,itime)           3  0.18052887739468537
 wait from XGC (stage, time,itime)          3  0.18052887739468537
 send to XGC(stage, time,itime)           4  0.18052887739468537
 wait from XGC (stage, time,itime)          4  0.18052887739468537
 send to XGC(stage, time,itime)           1  0.19055825947216790
 wait from XGC (stage, time,itime)          1  0.19055825947216790
    0.190558
  1.0389E-21  1.5528E-23  4.0477E-23  7.1409E-23  1.3878E-22  0.0000E+00  2
 send to XGC(stage, time,itime)           2  0.19055825947216790
 wait from XGC (stage, time,itime)          2  0.19055825947216790
 send to XGC(stage, time,itime)           3  0.19055825947216790
 wait from XGC (stage, time,itime)          3  0.19055825947216790
 send to XGC(stage, time,itime)           4  0.19055825947216790
 wait from XGC (stage, time,itime)          4  0.19055825947216790
 send to XGC(stage, time,itime)           1  0.20058764154965042
 wait from XGC (stage, time,itime)          1  0.20058764154965042
    0.200588
  1.0359E-21  1.6770E-23  4.4467E-23  7.2228E-23  1.3846E-22  0.0000E+00  2
 send to XGC(stage, time,itime)           2  0.20058764154965042
 wait from XGC (stage, time,itime)          2  0.20058764154965042
 send to XGC(stage, time,itime)           3  0.20058764154965042
 wait from XGC (stage, time,itime)          3  0.20058764154965042
```

bash

```
XGC_SZ_compression_and_Z-Check
    double      dpot[263243, 8] :
Rank 0: allocate 16.0672 MB for input buffer
[ZC] Reading ZC configuration file (zc.config) ...

File info:
  current step:   19
  last step:      19
  # of variables: 4:
Get info on variable 0: nnode
    integer     nnode      scalar
Get info on variable 1: nphi
    integer     nphi       scalar
Get info on variable 2: iphi
    integer     iphi       scalar
Get info on variable 3: dpot
    double      dpot[263243, 8] :
Rank 0: allocate 16.0672 MB for input buffer
[ZC] Reading ZC configuration file (zc.config) ...

File info:
  current step:   20
  last step:      20
  # of variables: 4:
Get info on variable 0: nnode
    integer     nnode      scalar
Get info on variable 1: nphi
    integer     nphi       scalar
Get info on variable 2: iphi
    integer     iphi       scalar
Get info on variable 3: dpot
    double      dpot[263243, 8] :
Rank 0: allocate 16.0672 MB for input buffer
[ZC] Reading ZC configuration file (zc.config) ...
```

bash

```
./XGC/timing/timing_000000020.txt.0000
                                       On   Called  Recurse     Wallclock
 MAIN_LOOP                              -      20      -      360.145721
   IPC_LOOP                             -      80      -      360.010529
     CHARGEI                            -      80      -      114.752220
       CHARGEI_SEARCH_INDEX             -      80      -        1.439226
       CHARGEI_SCATTER                  -      80      -       14.544598
       CHARGEI_GYRO_AVG                 -      80      -       31.085970
       CHARGEI_GA_RED_BCAST             -      80      -        1.561477
     POISSON                            -      80      -       81.726730
       POISSON_00MODE                   -      80      -       42.207512
       POISSON_TURB                     -      80      -       33.764774
       POISSON_SR_POT                   -      80      -        5.559200
     CCE_SEND_FIELD                     -      80      -        0.614316
     CCE_RECEIVE_FIELD                  -      80      -        0.050085
     CCE_PROCESS_FIELD                  -      80      -        0.000067
     GET_POT_GRAD                       -      80      -      128.205933
       GET_POT_LOOPS                    -      80      -        8.299930
       GET_POT_CNVRT                    -     160      -       10.837210
       GET_POT_SR                       -      80      -        2.538783
       GET_POT_MAT_MULT                 -      80      -        7.989079
       GET_POT_BCAST                    -      80      -       45.656815
       GET_POT_IDX_REORD                -      80      -       48.388771
     PUSH_I                             -      80      -        1.076356
       PUSH_LOOP                        -      80      -        1.068416
     DIAGNOSIS                          -      80      -       12.849648
       DIAG_1D_OUTPUT                   -      20      -        2.915997
       DIAG_3D                          -      20      -        9.927645
     SHIFT_I                            -      80      -        8.998302
       MEM_CLEAN_I                      -      80      -        0.008543
   COLLISION                            -      20      -        0.000061

head sum =    0.00112 wallclock seconds
```

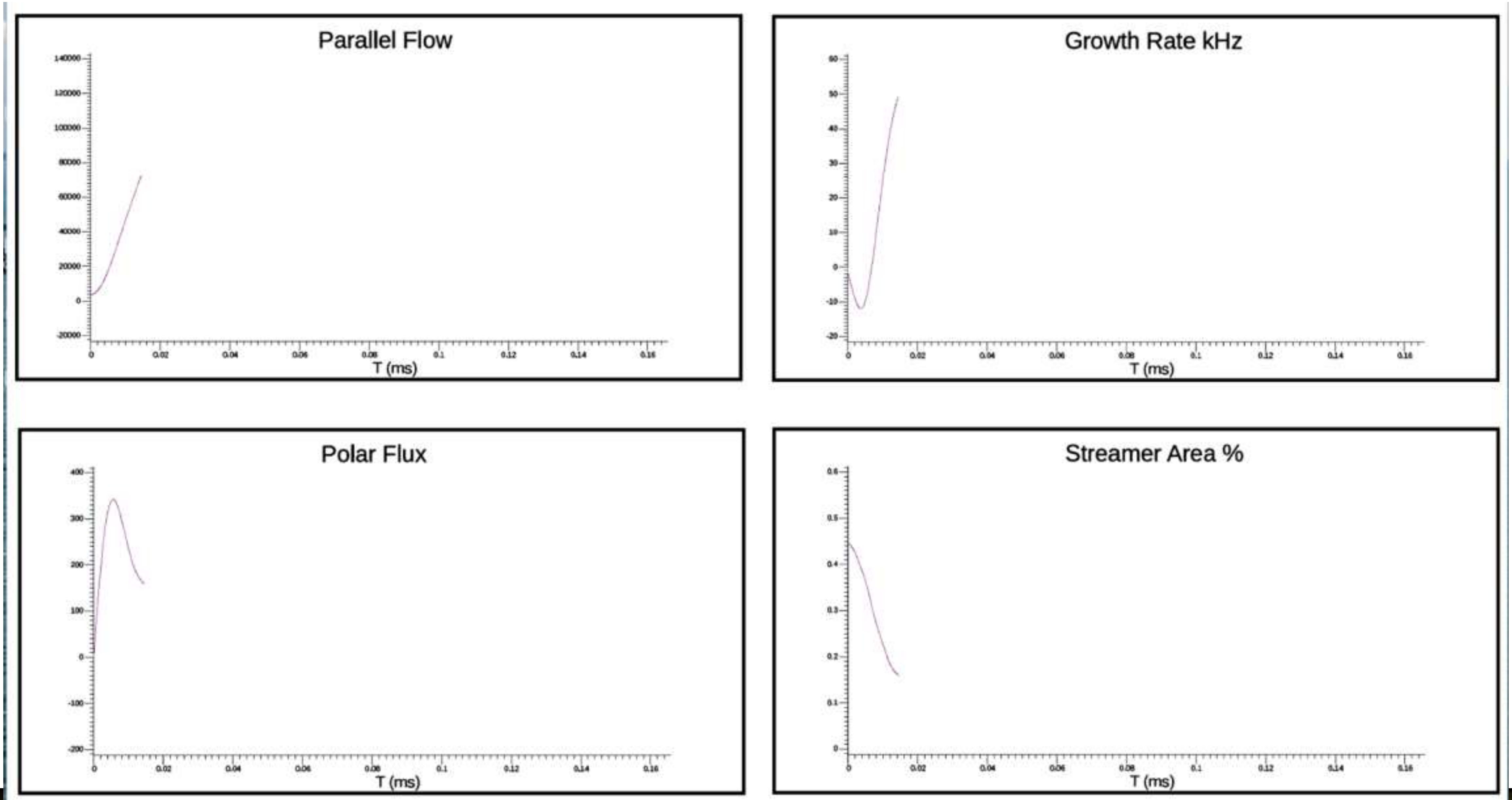
bash

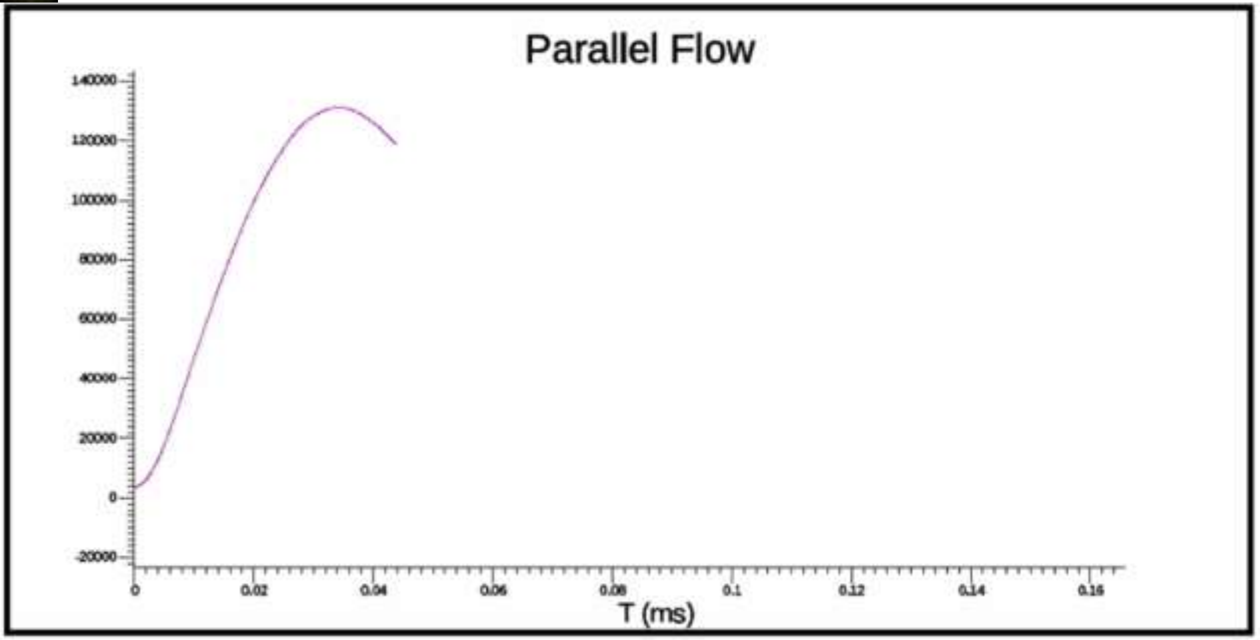
```
VTKM
  CSE: 6
  CSE: 7
Streamer Energy = 1.09693e-13
Total Energy    = 1.11664e-13
Compute Streamer Energy :: Wall Time = 2158.99 ms

CSA: 0
CSA: 1
CSA: 2
CSA: 3
CSA: 4
CSA: 5
CSA: 6
CSA: 7
Compute Streamer Area :: Wall Time = 48.9873 ms

Compute Growth :: Wall Time = 5.73621 ms

Threshold: -100 0.276
Threshold: 0.467 100
Threshold: 0.276 0.467
Threshold: -3.27663e-10 -3.27663e-11
Threshold: 3.27921e-11 3.27921e-10
3D Views :: Wall Time = 8325.63 ms

Compute Ballooning :: Wall Time = 3365.62 ms

  CSE: 0
  CSE: 1
  CSE: 2
  CSE: 3
  CSE: 4
  CSE: 5
```

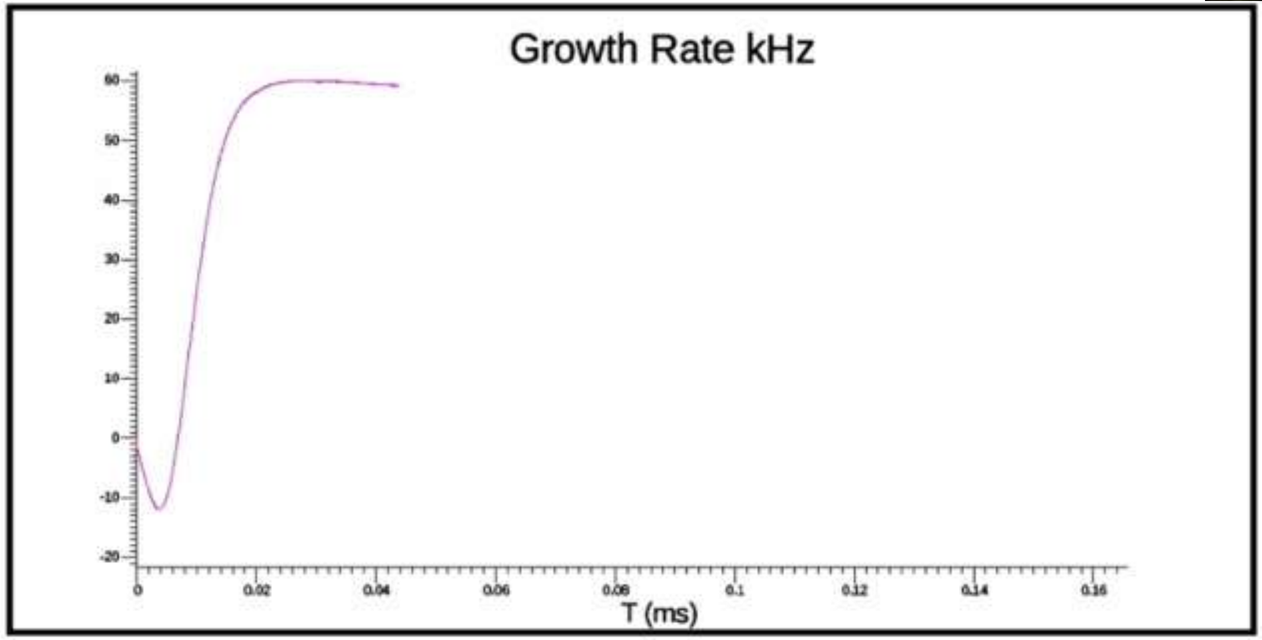
bash

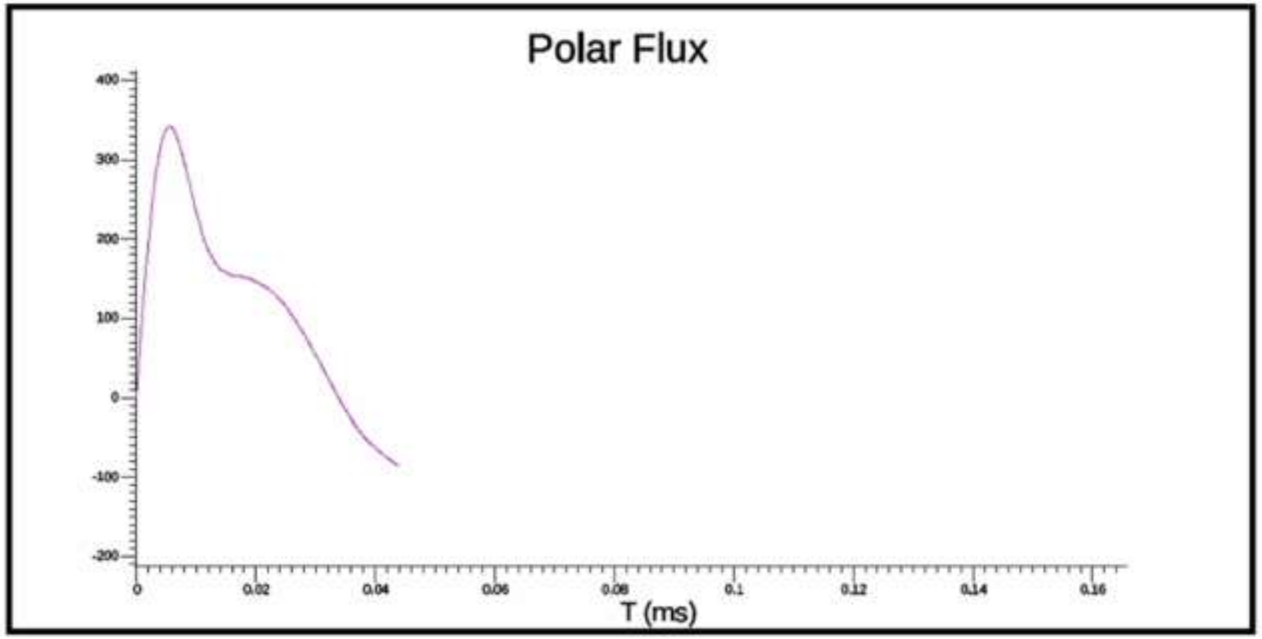
```
GENE_MGARD_compression_and_Z-Check

File info:
  current step:   20
  last step:      20
  # of variables: 10:
Get info on variable 0: nx
    integer     nx         scalar
Get info on variable 1: nz
    integer     nz         scalar
Get info on variable 2: gx
    integer     gx         scalar
Get info on variable 3: gz
    integer     gz         scalar
Get info on variable 4: ox
    integer     ox         scalar
Get info on variable 5: oz
    integer     oz         scalar
Get info on variable 6: time
    double      time       scalar
Get info on variable 7: timestep
    integer     timestep scalar
Get info on variable 8: phi_real
    double      phi_real[24, 160] :
Get info on variable 9: phi_imag
    double      phi_imag[24, 160] :
Rank 0: allocate 0.0294189 MB for input buffer
[ZC] Reading ZC configuration file (zc.config) ...
Got: 4    2e-05
Tog: 17           129
[ZC] Reading ZC configuration file (zc.config) ...
Got: 4    2e-05
Tog: 17           129
```

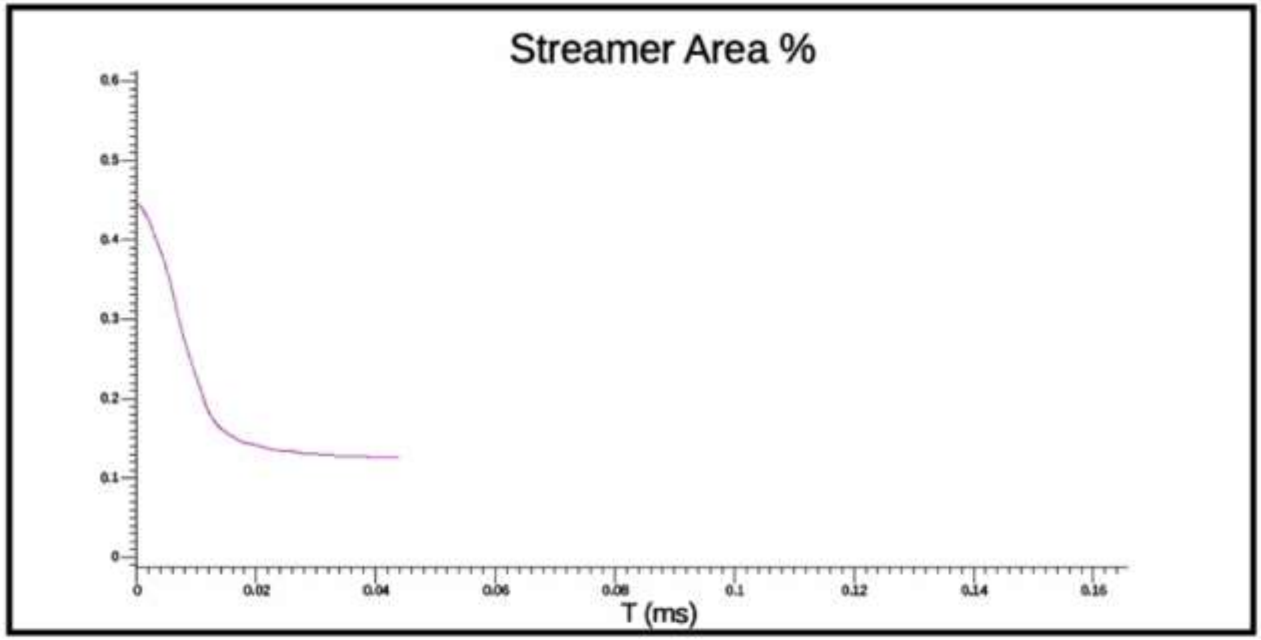
Desktop 2: The physics screens to monitor





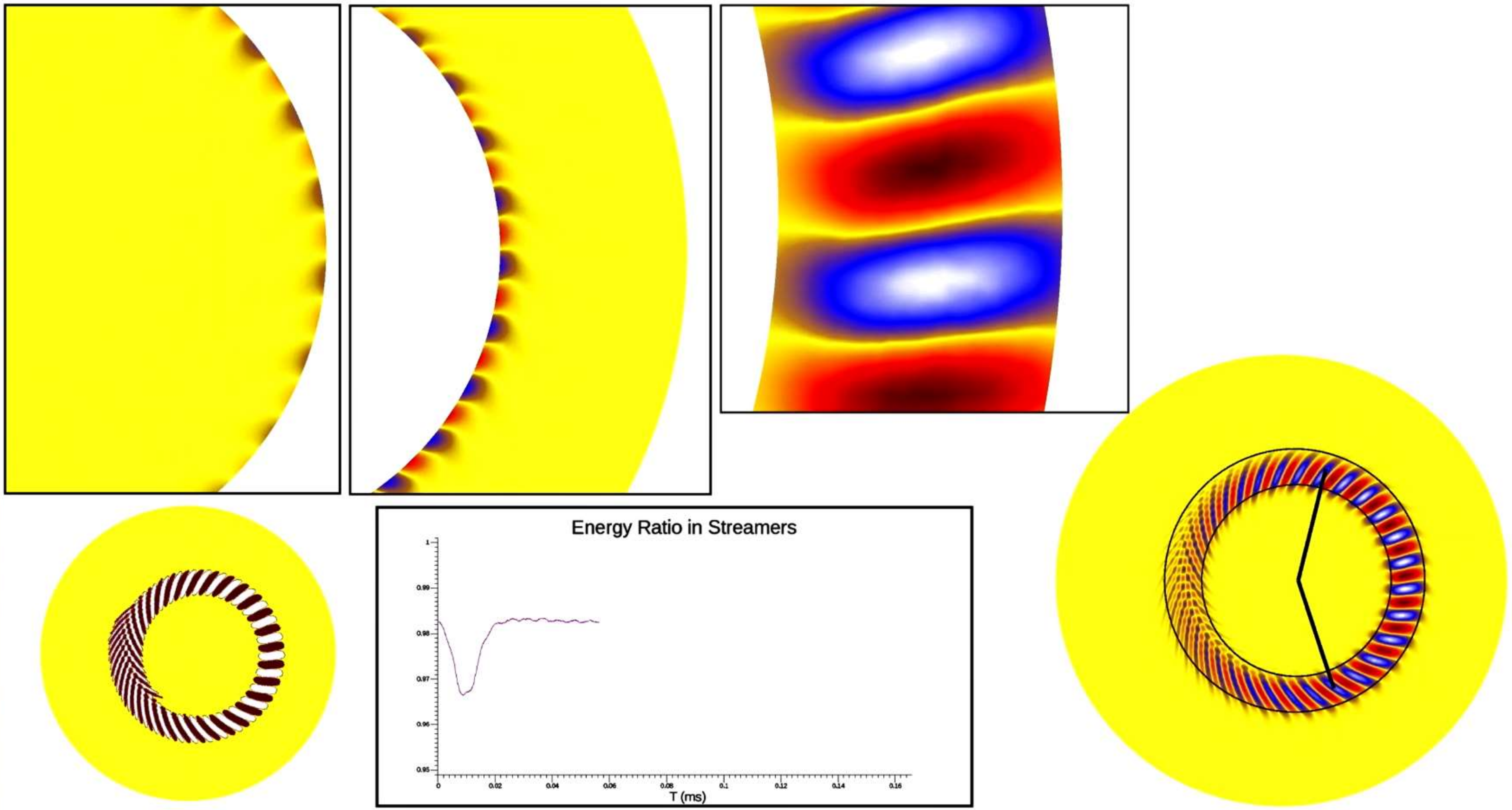


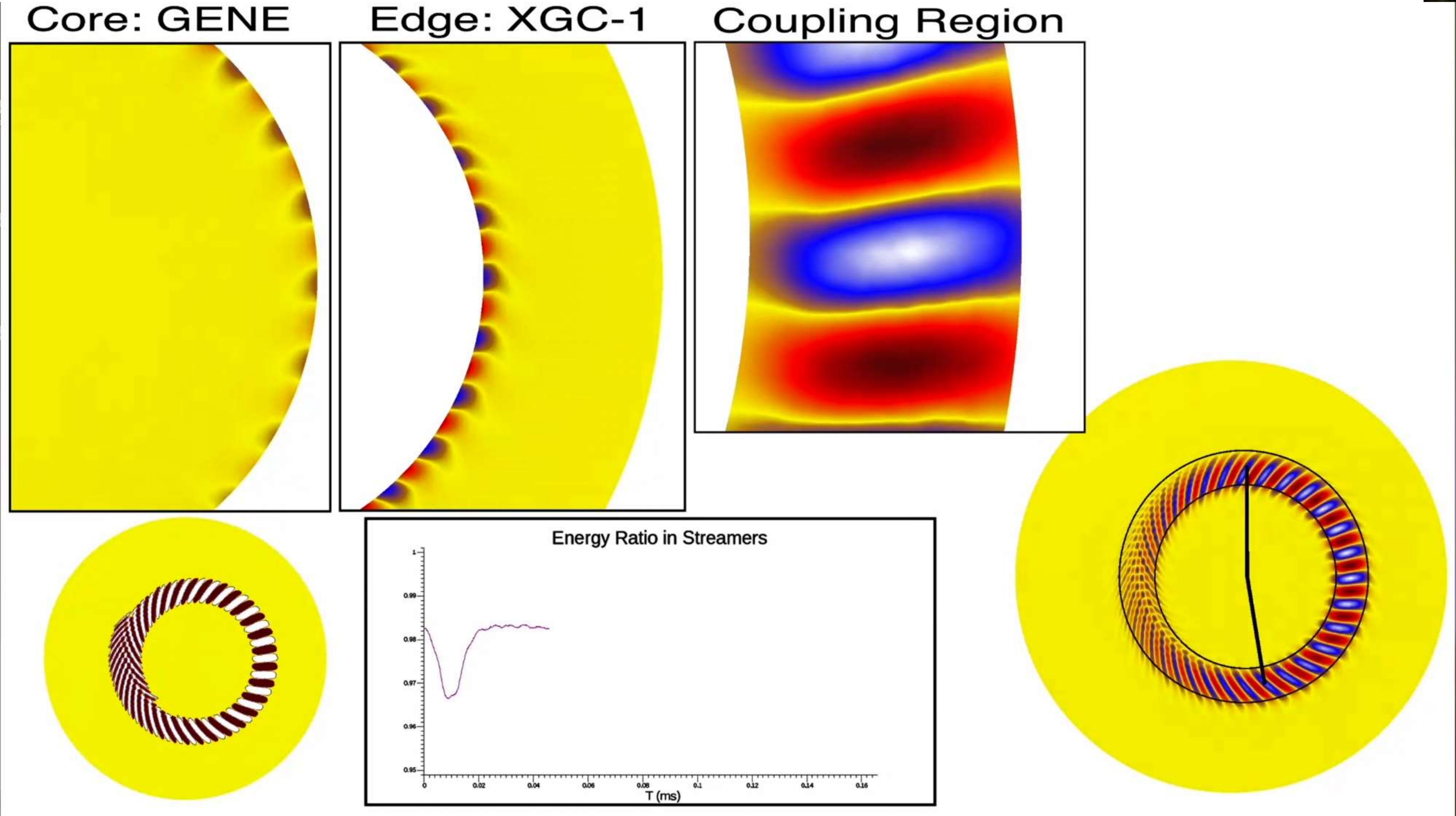


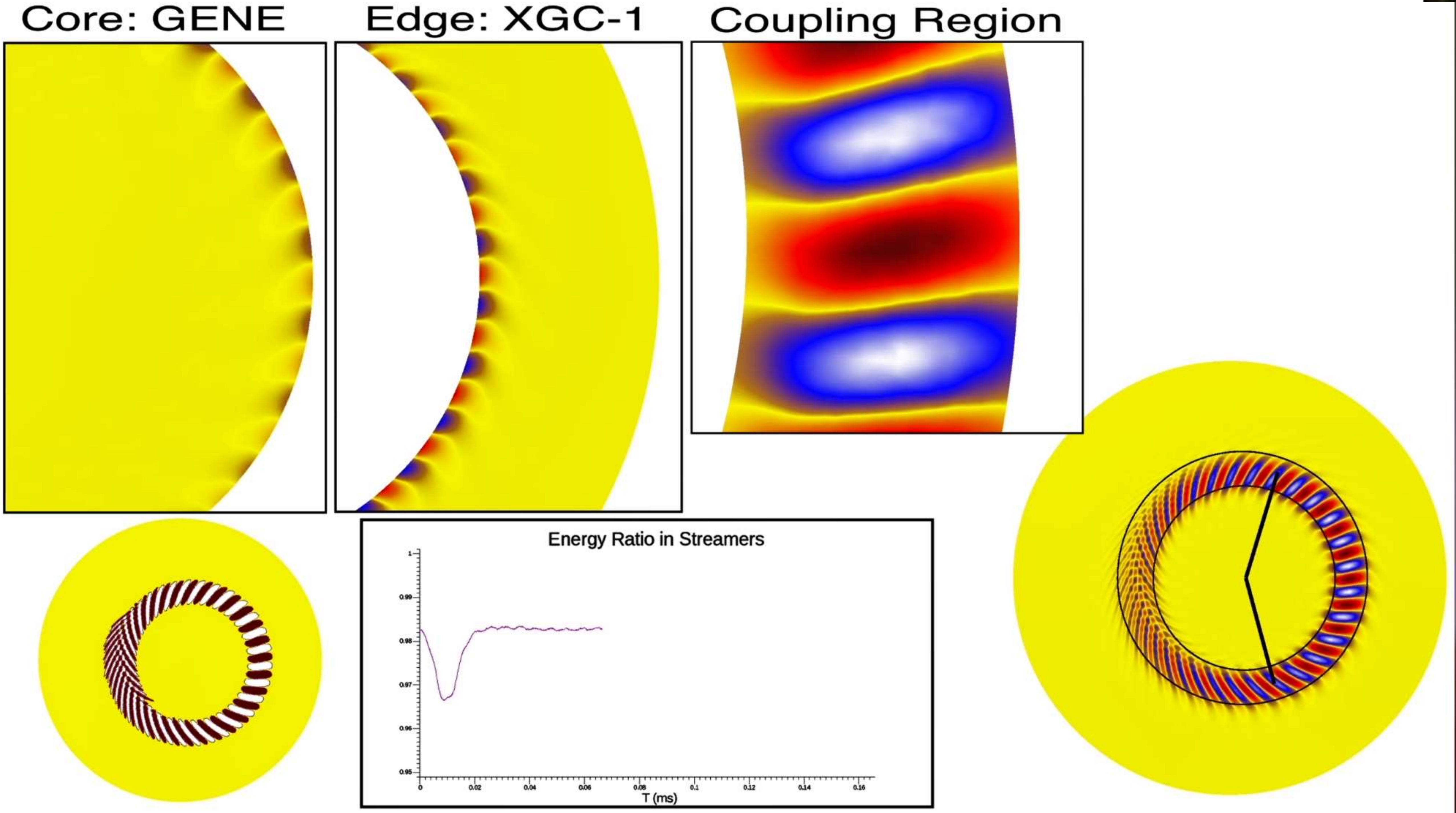


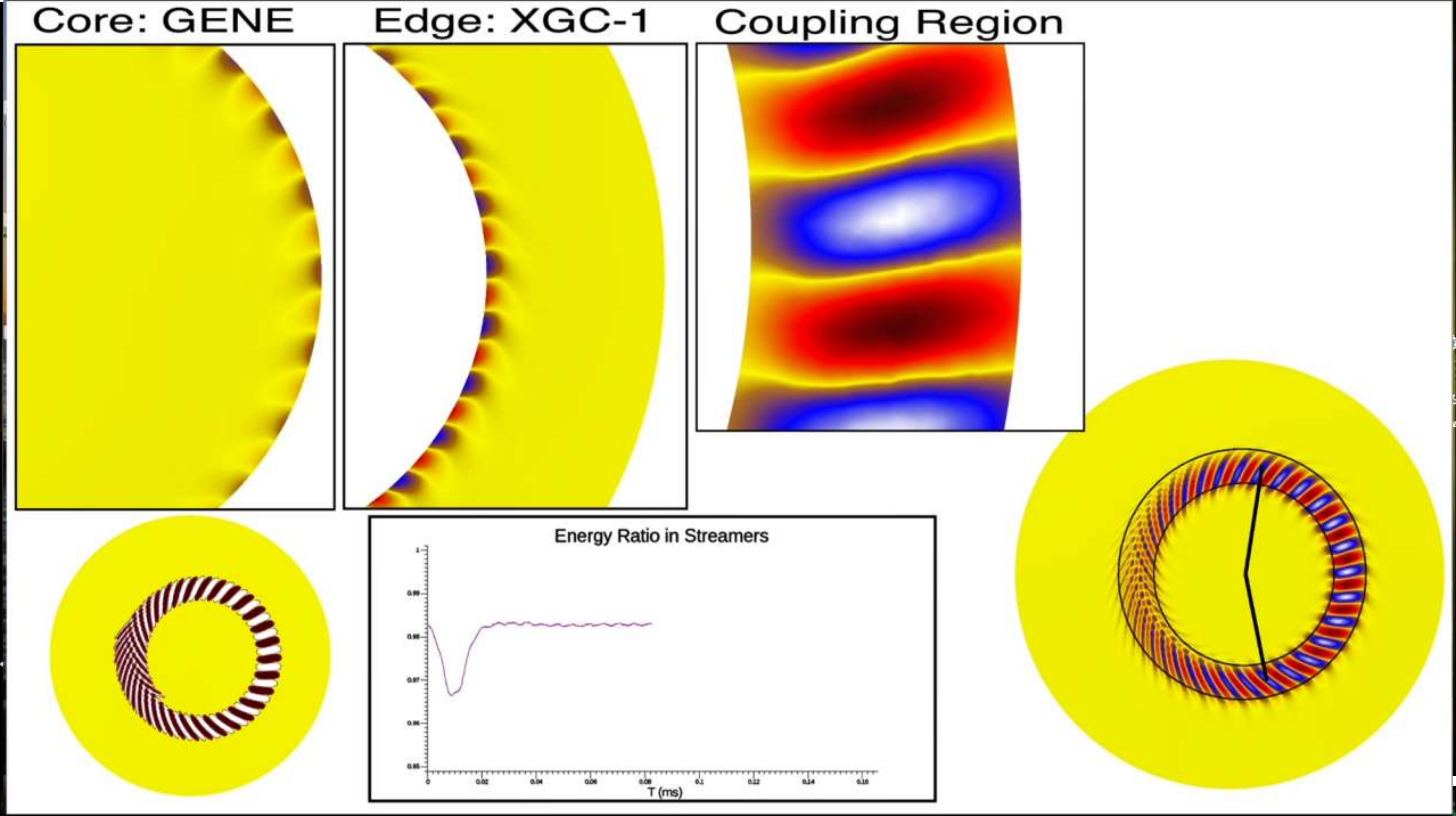
klasky@ornl.gov

Desktop 3: in situ visualization & feature tracking



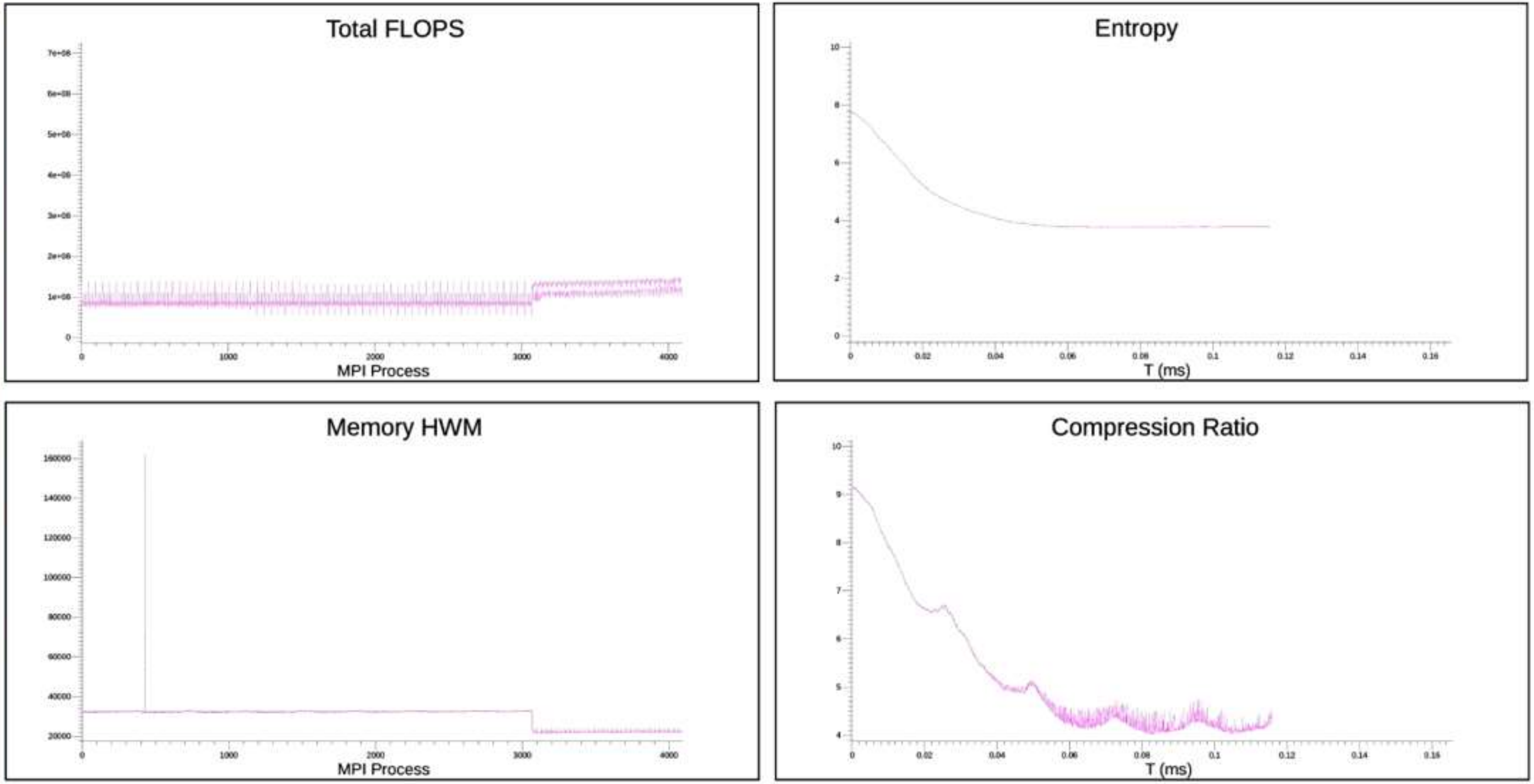

Core: GENE
Edge: XGC-1
Coupling Region
Energy Ratio in Streamers
1
0.99
0.98
0.97
0.96
0.95
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
T (ms)

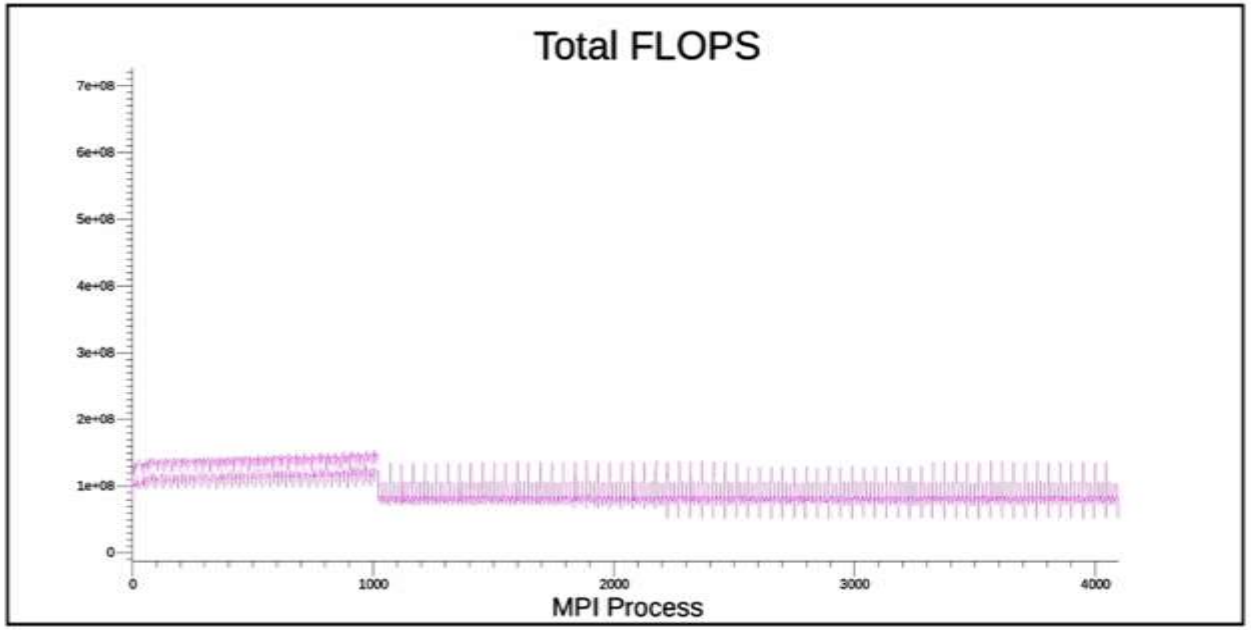
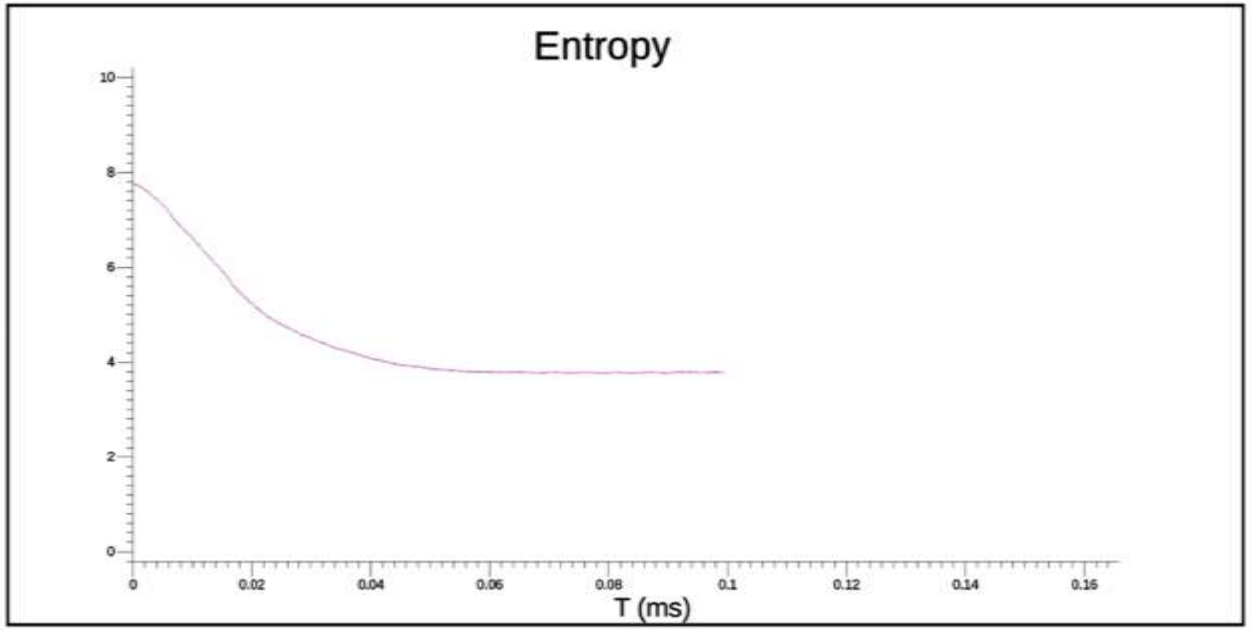
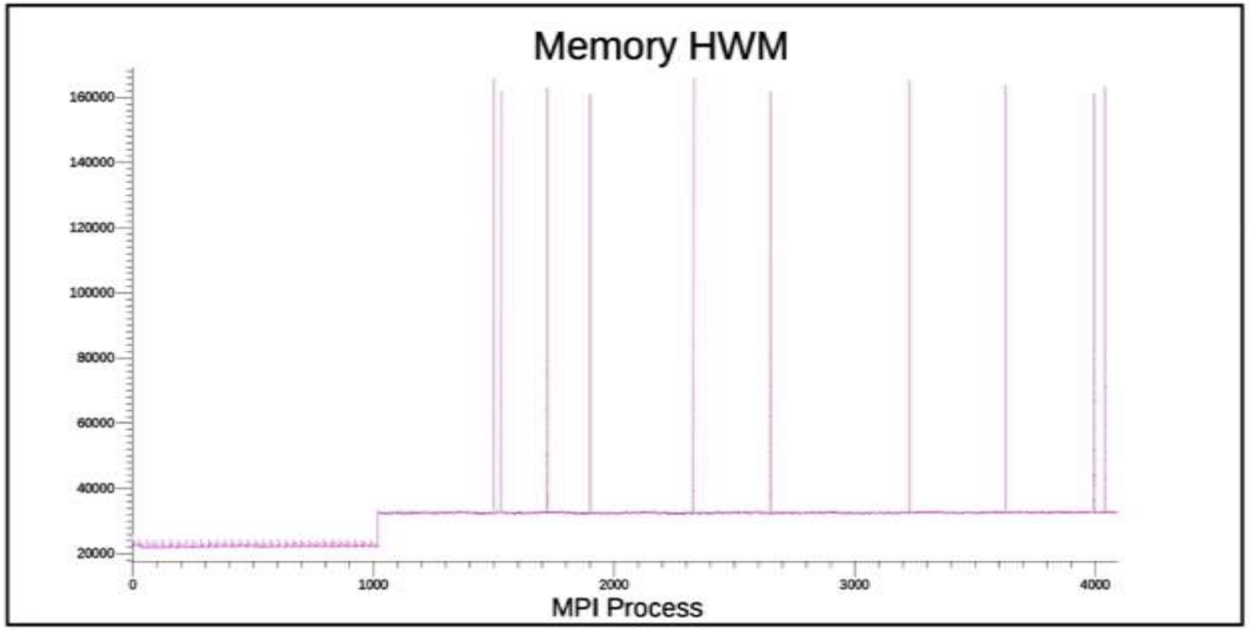
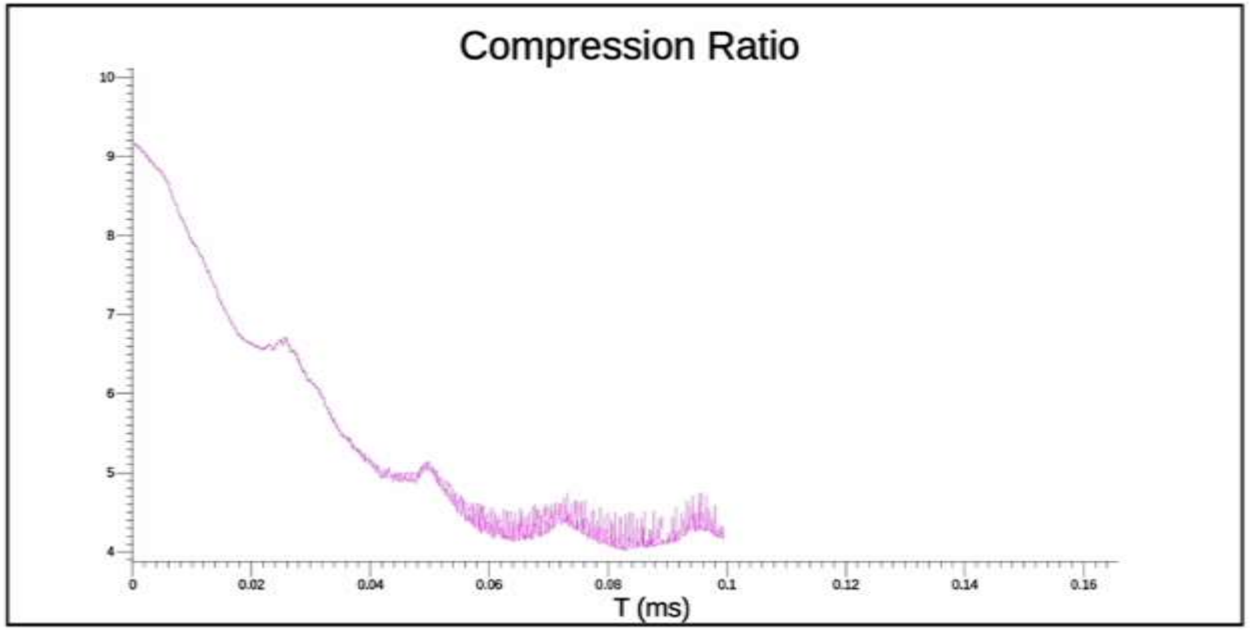

Core: GENE
Edge: XGC-1
Coupling Region
Energy Ratio in Streamers
1
0.99
0.98
0.97
0.96
0.95
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
T (ms)

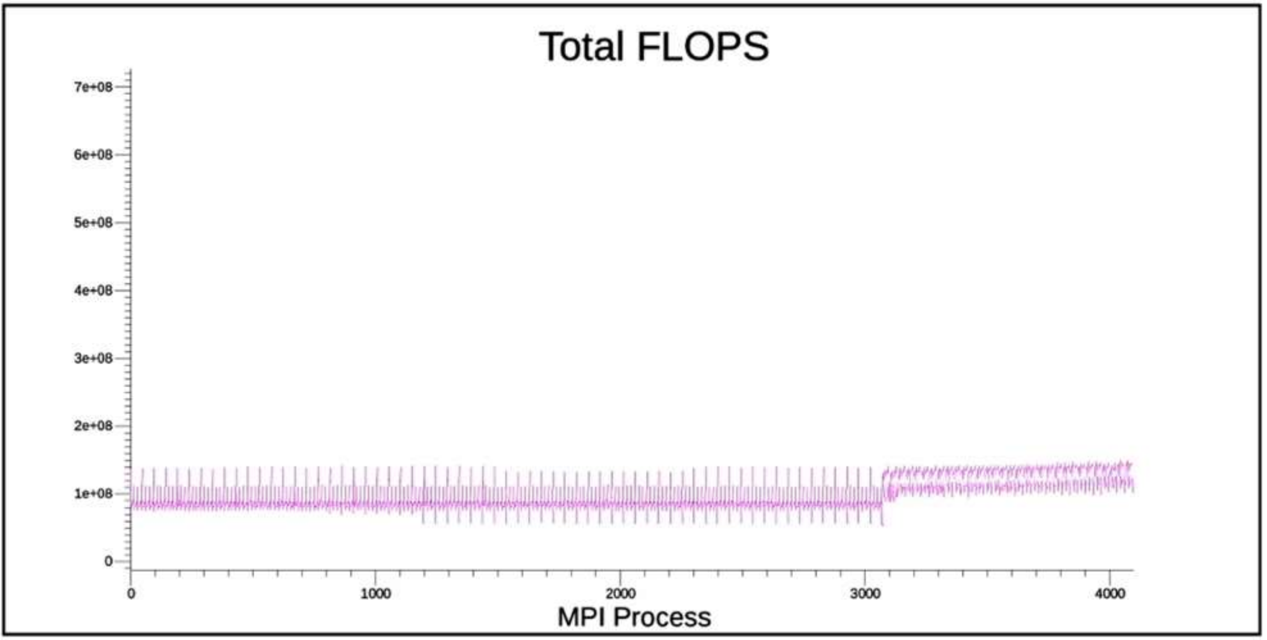


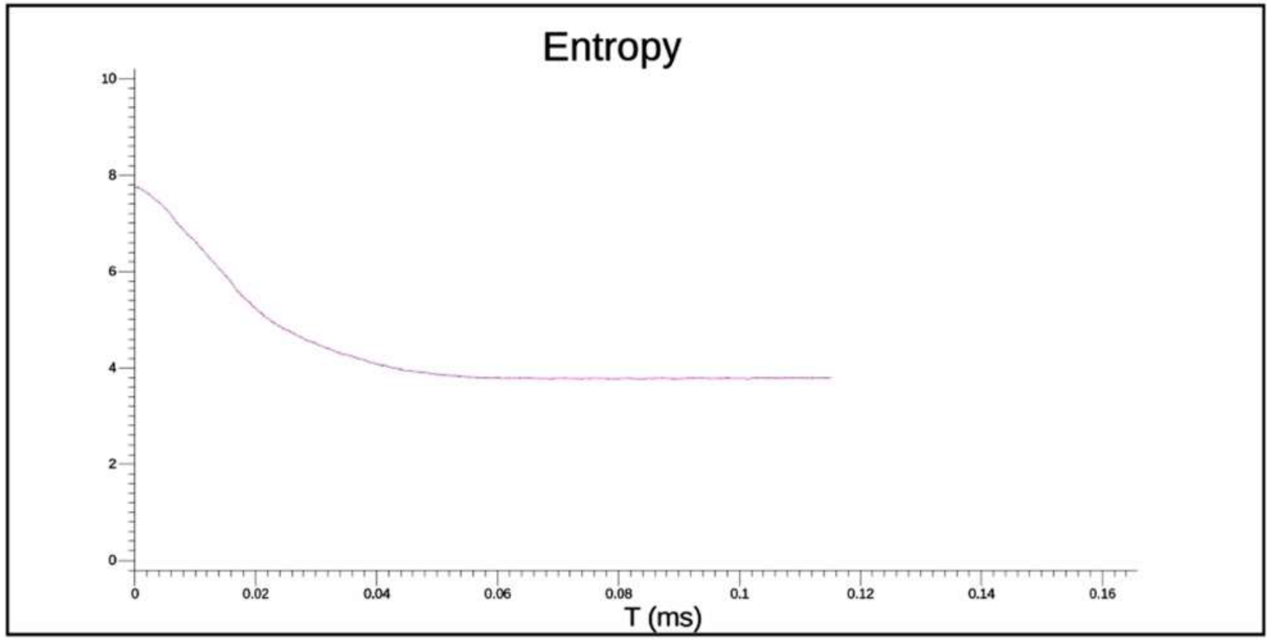
Core: GENE
Edge: XGC-1
Coupling Region
Energy Ratio in Streamers
1
0.97
0.85
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
T (ms)

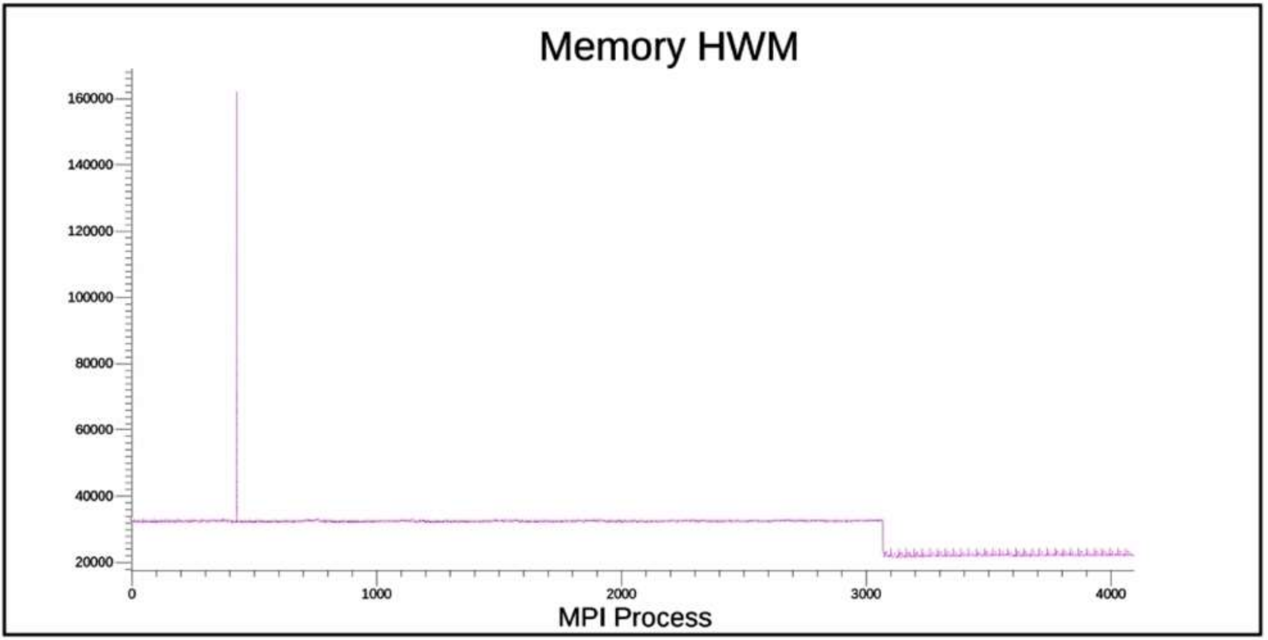
Desktop 4: Performance & errors

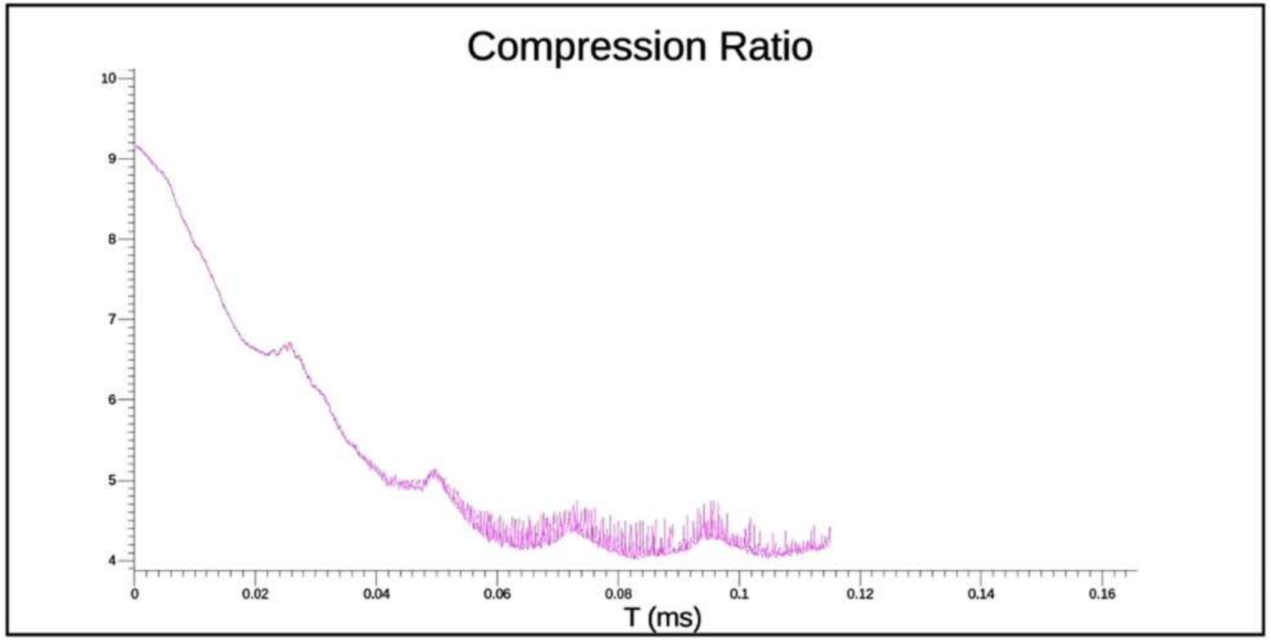


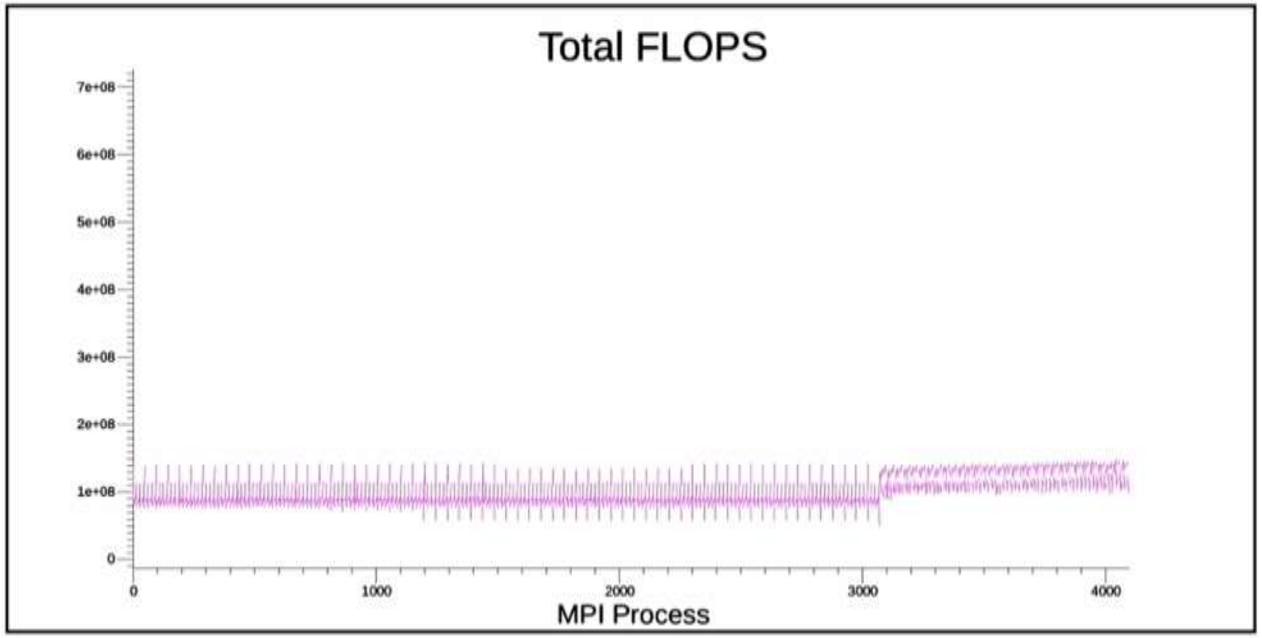

Total FLOPS
MPI Process

Entropy
T (ms)

Memory HWM
MPI Process

Compression Ratio
T (ms)

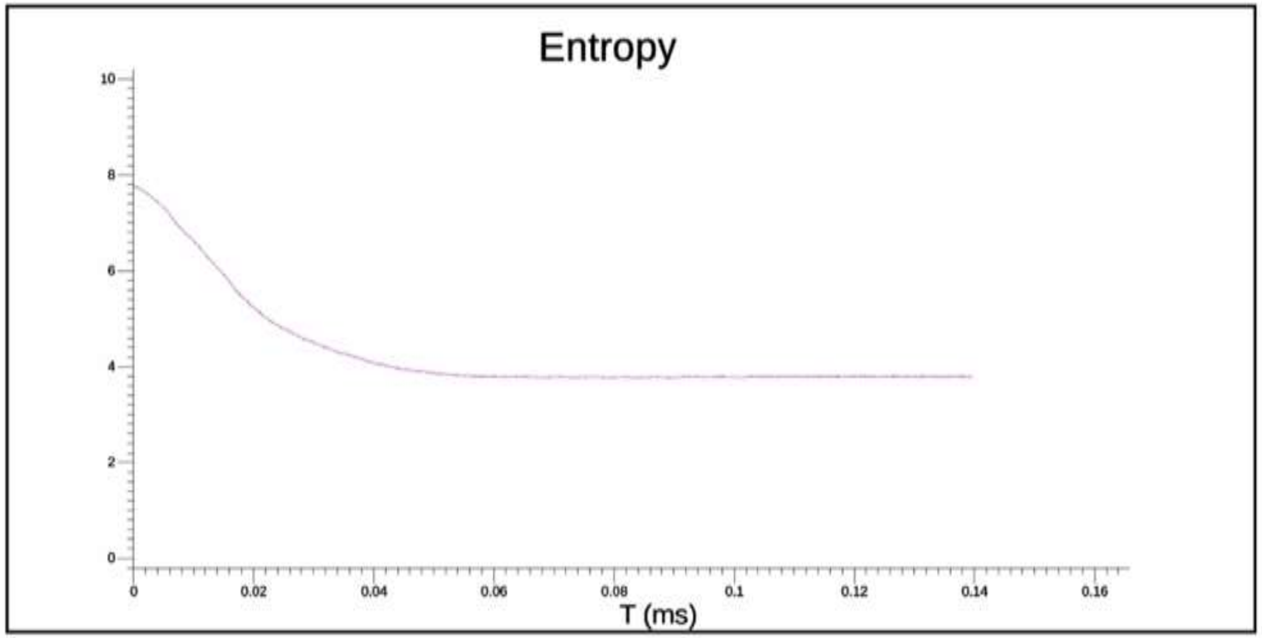

Total FLOPS
7e+08
6e+08
5e+08
4e+08
3e+08
2e+08
1e+08
0
0
1000
2000
3000
4000
MPI Process

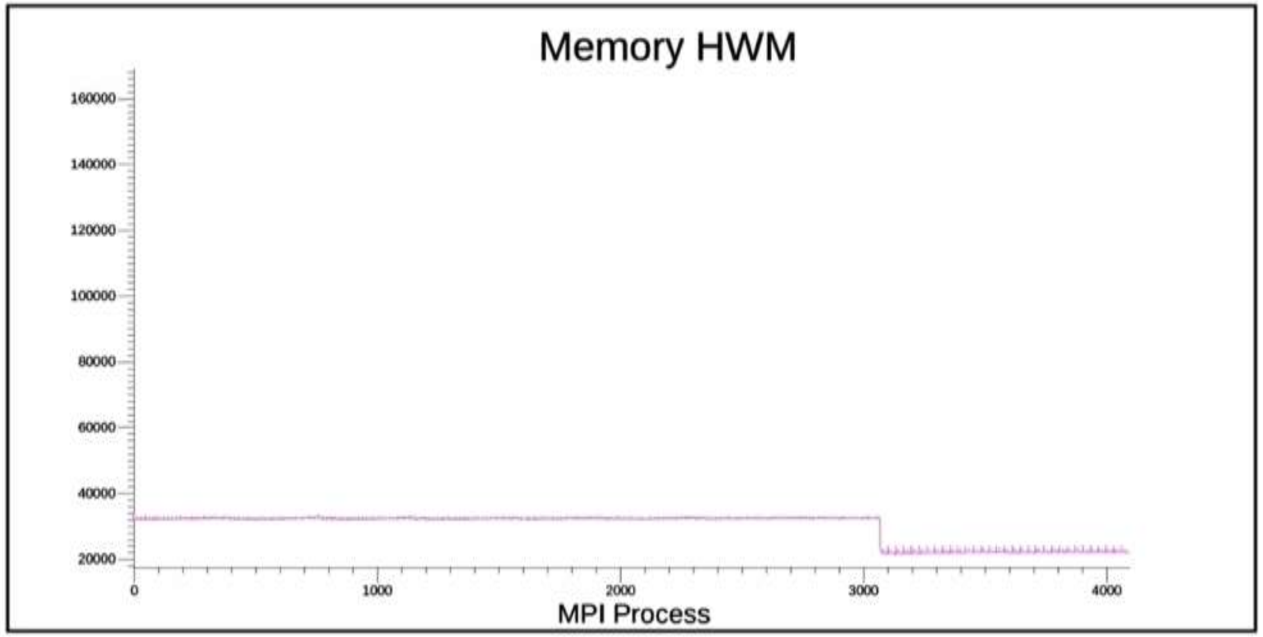

Entropy
10
8
6
4
2
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
T (ms)

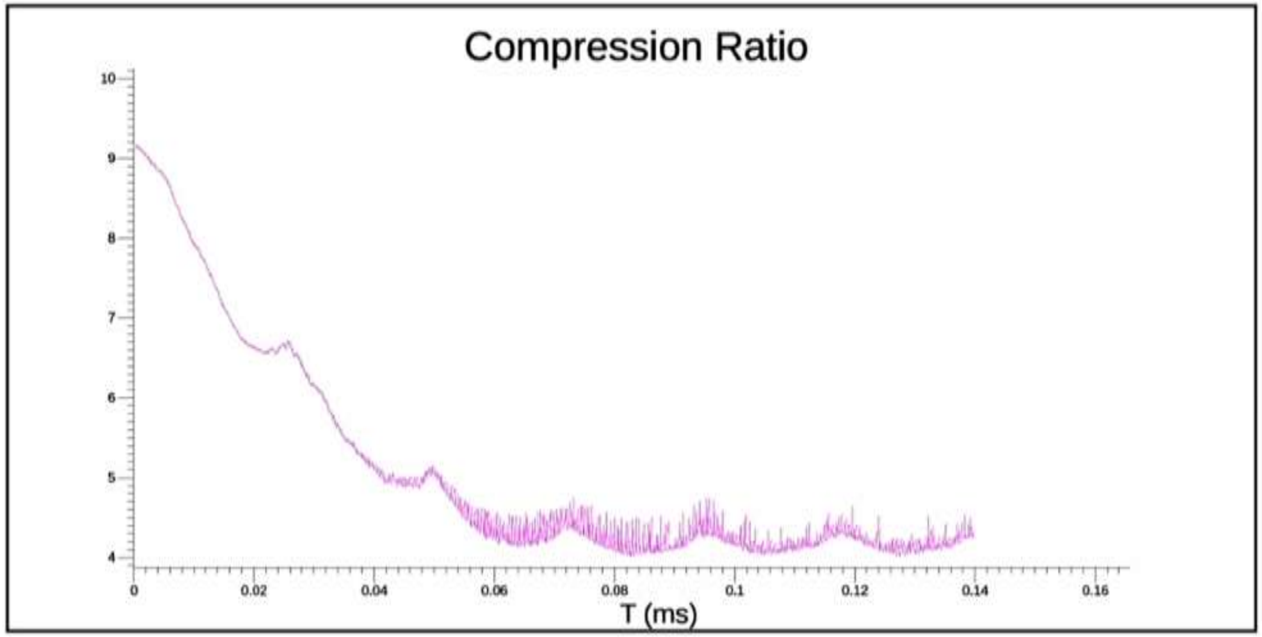

Memory HWM
160000
140000
120000
100000
80000
60000
40000
20000
0
1000
2000
3000
4000
MPI Process


Compression Ratio
10
9
8
7
6
5
4
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
T (ms)


Total FLOPS
7e+08
6e+08
5e+08
4e+08
3e+08
2e+08
1e+08
0
0
1000
2000
3000
4000
MPI Process


Entropy
10
8
6
4
2
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
T (ms)


Memory HWM
160000
140000
120000
100000
80000
60000
40000
20000
0
1000
2000
3000
4000
MPI Process


Compression Ratio
10
9
8
7
6
5
4
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
T (ms)

Conclusions

- ADIOS was created to
  - Reduce the time spent reading and writing large data on HPC resources → Large data I/O & storage
  - Process data from either streams or files, using a "publish-subscribe mechanism" → publish - subscribe
  - Provide developers an extendible framework to place the best software for I/O into one framework → SDK
  - Provide users the ability to couple codes together efficiently on large numbers of processors (on-node, off-node, off-machine) → Staging
  - Provide the ability to query data both on-line & offline → Queries
  - Provide a place to reduce data in "buckets" to move data progressively according to information content → Refactoring

- From app partnerships & Research
  - PiconGPU, … (IPDPS 2009, 159), HPDC 2011, 75)
  - GTC,… (IPDPS 2010, 162)
  - PiconGPU, SKA, (Concurrency , 74)
  - S3D, XGC (Cluster 2012, 151)
  - Impact (Euro-Par 2011 , 99)
  - XGC (SIAM SciComp 2018)

Questions