Visualization and Analysis Requirements for In Situ Processing for a Large-Scale Fusion Simulation Code

James Kress
University of Oregon &
Oak Ridge National Laboratory
kressjm@ornl.gov

David Pugmire
& Scott Klasky
Oak Ridge National Laboratory
pugmire@ornl.gov, klasky@ornl.gov

Hank Childs
University of Oregon &
Lawrence Berkeley National Laboratory
hank@cs.uoregon.edu

***Abstract*—In situ techniques have become a very active research area since they have been shown to be an effective way to combat the issues associated with the ever growing gap between computation and I/O bandwidth. In order to take full advantage of in situ techniques with a large-scale simulation code, it is critical to understand the breadth and depth of its analysis requirements. In this paper, we present the results of a survey done with members of the XGC1 fusion simulation code team in order to gather their requirements for analysis and visualization. We look at these requirements from the perspective of in situ processing and present a list of XGC1 analysis tasks performed by its physicists, engineers, and visualization specialists. This analysis of the specific needs and use cases of a single code is important in understanding the nature of the needs that simulations have in terms of data movement and usage for visualization and analysis, now and in the future.**

1. Introduction

Current trends in supercomputing point to a future where increases in core counts are greatly outpacing increases in memory and I/O bandwidth. These systems will make it possible to compute far more data than can regularly be moved to disk. As a result, the vast majority of data produced by simulations will be lost, or the workflow will stall under the burden of I/O [1]. Simulation scientists are faced with the problem of deciding what small fraction of data can be saved, and what must be discarded. Ever lurking within these decisions is the possibility of lost scientific knowledge.

Research efforts for efficiently using these systems are following several paths. These paths include more efficient use of the memory hierarchy in terms of I/O [2], [3], [4] and burst-buffers [5], [6], data compression and subsetting [7], [8], [9], [10], frameworks that efficiently use the available compute cores to process data [11], [12], [13], and in situ visualization and analysis methods [14], [15], [16], [17].

In this short paper we consider the area of in situ visualization methods. We focus our efforts on a study of the XGC1 [18] scientific team, and the workflows being run on leading edge supercomputing systems. We present a survey of the predominant visualization and analysis tasks in this workflow, and, for each, describe how the task is currently performed given a list of computational, time, and resource constraints. We believe this study of the XGC1 project is valuable, since it formalizes the specifics of in situ requirements for a simulation code for later usage by visualization scientists. While a subset of this information is available in several research papers, we think a study dedicated exclusively to cataloging requirements gives a more complete picture. This information could in turn be used for engineering software designs, hardware designs, and conducting feasibility studies.

We know of no efforts to provide a formalized way to approach in situ visualization given the computational and data constraints and requirements of a particular simulation. Such a formalization would provide a framework to reason about the time required for input and output on a particular computing system, along with the scientific requirements for visualization in a workflow, which in turn informs the feasibility of that in situ task. While we do not solve the feasibility problem in this work, we believe that data gathered in this work will be input to solutions for the feasibility question.

In the remainder of this short paper, we discuss related works in Section 2, describe the XGC1 project and its output data and data sizes in Section 3, and describe visualization and analysis requirements for XGC1 from our interview process in Section 4.

2. Related Work

The Advanced Scientific Computing Research (ASCR) Scientific Grand Challenges Workshop Series produced a series of reports spanning eight different scientific domains (High Energy Physics, Climate, Nuclear Physics, Fusion, Nuclear Energy, Basic Energy Sciences, Biology, National Security) [19], [20], [21], [22], [23], [24], [25], [26], as well as a summarization and exposition report that further explored the visualization and analysis requirements from the eight different reports as a whole [1]. Each of these reports singles out data movement, storage, and analysis as a major obstacle in the move to exascale. Many of these scientific domains will be required to deal with petabytes, or

even exabytes of data over the course of a simulation. Some even expect to need to introduce new data representations and types. These looming problems make it important to start understanding the entire workflows of these codes, and to develop methods and infrastructure that will enable them to perform on future architectures. While these reports accomplish the goal of setting a research agenda around visualizing large data, they do not outline an approach for scientists to reason out their visualization needs based on data size and complexity, computational resources, or algorithm complexity, which contrasts with our study.

We know of no work focusing specifically on cataloging and categorizing the different visualization and analysis tasks of a simulation code. There are however instances of visualization and analysis requirements being reported in conjunction with a study.

A work by Bennett et al. [27] reports on a use case with combustion simulations using S3D, where features are tracked, identified, and visualized both in situ and in transit. Their work utilized in situ and in transit methods using a volume of nearly 1 billion cells and 16 seconds average wall time per time step using 4896 cores.

Pugmire et al. [9] explore a feature tracking and identification use case in the XGC1 simulation code, using a data set of nearly 1 billion particles and a time budget of 10 seconds per simulation time step. In this work, the authors describe a system that intelligently handles the tracking of particles and features of a simulation in real time, in a user specified area of interest.

Ellsworth et al. [28] describe a time-critical pipeline for weather forecasting using the GEOS4 simulation code. This code is run under very tight time constraints four times a day which requires the visualization to be performed with minimal overhead. The visualization was performed on data consisting of 23 million cells with up to seven 3D and four 2D fields per cell.

Malakar et al. [29] describe a series of visualization tasks done with the LAMMPS simulation code. The data contained 1 billion atoms, using 91 GB per simulation time step. Typical runs consisted of 1000 time steps, with output every 100 time steps.

Slawinska et al. [30] demonstrate the incorporation of ADIOS into Maya for an astrophysics simulation workflow. Using in situ techniques, they reduced the amount of data needed to perform their visualization and analysis task from 4.5 TB down to 24 GB that would normally be saved to disk without in situ.

From these past works we have been able to get a sense of some of the data sizes and visualization and analysis requirements from other large-scale simulation codes. None of these reports however gives a full picture of the data and analysis requirements stemming from these simulation codes. Without understanding both the breadth and depth of the needs of these codes in terms of data movement and usage, future research efforts on in situ techniques may miss an important aspect or problem that is very important to large-scale simulation codes, but just has not been formally presented to the community.

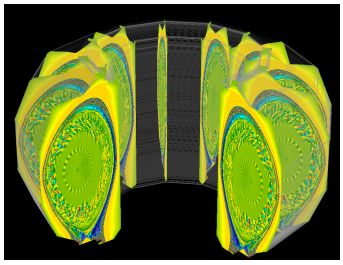


Figure 1: Example of an XGC1 mesh with planes equally spaced around the central axis of the tokamak.

3. XGC1 Project

XGC1 is a 5D gyrokinetic ion-electron particle in cell (PIC) code used to study fusion of magnetically confined burning plasmas. XGC1 is used in particular to study the turbulent region on the outer region of the plasma called the *edge*. The simulation proceeds by computing the interactions of a very large number of particles, and then depositing the particles onto a finite element mesh. The mesh, as shown in Figure 1, consists of a number of 2D planes positioned uniformly around the toroidal shape of the tokamak. The number of planes used, typically between 16 and 64, is specified by the scientists to capture the expected waveform distributions. The particles, which interact within the toroidal space of the mesh, are statistically deposited onto the mesh. This deposition step provides a statistical view of simulation, as well as helps optimize the simulation runtime.

XGC1 scientists typically run two different sizes of simulations, which we categorize as *medium* and *large.* These run sizes are defined by three factors (1) the number of compute processes; (2) the number of particles per process; and (3) the number of nodes in the mesh. These factors are quantified for the medium and large runs in Table 1.

TABLE 1: Simulation size characteristics, particle counts, and wall time per simulation time step for two different XGC1 run sizes.

	Medium Run	Large Run
Number of Processes	65,536	262,144
Number of Particles Per Process	100,000	500,000
Number of Mesh Nodes	100,000	1,069,247
Average Wall Time Per Time Step	2-4 min	5-10 min

3.1. XGC1 Output Data Types and Sizes

In this section we discuss the variety of outputs produced by XGC1, with an emphasis on outputs most relevant for analysis and visualization.

The largest output file in XGC1 is the `restart` file, and contains the state of each particle at a particular time step. Medium and large runs will contain around 6 billion and 150 billion particles, respectively.

The second largest output file in XGC1 is the `restartf0` file, which is used for post-processing detection of abnormal particles. This file contains a mapping of

TABLE 2: A summary of the output data from XGC1 that is used most often by those interviewed. The table shows average sizes for medium and large runs, as well as how often the data changes.

	File Size (GB)		
File Name	**Medium Run**	**Large Run**	**Output Frequency**
restart	976	19,531	1-100 Time Steps
restartf0	48	522	1-100 Time Steps
mesh	0.025	0.256	Static
output.bfield	0.075	0.75	Static
oneddiag	0.002	0.03	Every Time Step
3d	0.075	0.8	Every Time Step
f3d	0.35	2.0	Every Time Step

each plane in the unstructured grid to a regular mapping in phase space. This mapping produces smooth contours for non-turbulent particles, making it easier to identify the non-smooth contours of turbulent particles.

The unstructured 3D mesh in XGC1 is described in the `mesh` file, which is static over time, and specifies the points and connectivity of a single plane, and the number of planes around the tokamak. Medium and large runs will use about 100K and 1M points per plane respectively.

The `output.bfield` file contains the steady state magnetic field defined on the unstructured mesh and is static.

The `oneddiag` file contains general diagnostics that are appended after each time step. This file contains around 80 different diagnostic values, such as densities, flow, and momentum values, and is used to calculate a number of derived quantities.

The `3d` file is produced every time step and contains data for each plane in the simulation. The data is partitioned based on the underlying triangular mesh describing the tokamak. That is, this data is produced during the deposition and data reduction step in the simulation, where raw particle data is deposited onto the triangular mesh, producing an average value for that mesh region.

The `f3d` file is produced every time step and consists of ion and electron information relating to temperature, density, and velocity. The data is partitioned just as in the `3d` case, and is based on the underlying triangular mesh describing the tokamak, resulting in an average value for each mesh region.

Table 2 contains a summary of the previously detailed information on XGC1 output files and associated file size.

4. XGC1 User Surveys

The XGC1 project is composed of a large membership, including physicists, experimentalists, analysts, and computer scientists. This diversity of backgrounds leads to a broad range of activities to be performed on various parts of the data, each requiring varying computational and data resources. In order to gain a holistic understanding of the project, we conducted interviews with 7 different XGC1 team members, covering key areas of the XGC1 workflow. Our interviews started with the same questions for each participant, although follow-on questions were adapted based on the interests and expertise of the participant. From these interviews we have distilled a list of required and "nice-to-have" analysis routines on XGC1 data. Finally, while our interest in these requirements is in how they apply to in situ processing, we note that in many cases they are applicable to post hoc processing requirements as well.

The required and nice-to-have analysis routines can generally be categorized into three areas: (1) visualization and analysis, (2) simulation monitoring, and (3) debugging and performance engineering. For each of these three areas we will report on our findings from our interviews, as well as indicate which of the items is a Data Analysis and Visualization (DAV) task. DAV's are specific instances of the requirements we identified through our interview process. One key finding from the interviews, which is highly relevant for in situ, is that XGC1 allows up to 10% of total simulation time to be devoted to I/O. This fact must be kept in mind as new data requirements and fidelities are output for visualization and analysis tasks. The requirements gathered from the XGC1 team in each of the three areas are presented in Sections 4.1, 4.2, and 4.3 respectively.

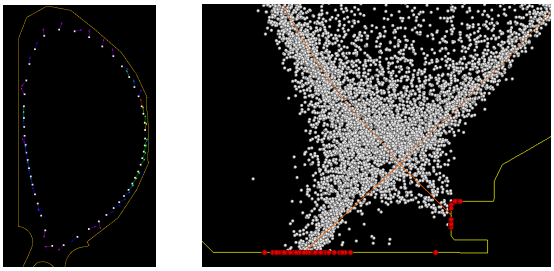
4.1. Visualization and Analysis

A common analysis task in XGC1 is to make an image of a feature or region of interest. Images can serve several distinct functions in XGC1: (1) a diagnostic tool for checking new physics in the code, (2) a debugging and verification mechanism for new visualization routines, and (3) a method of exploring, discovering, and understanding new properties in the tokamak that either were not known or have been assumed to exist by the physics community. There are two types of images needed from XGC1: static plots and videos of time varying quantities.

Make Static Plots. Static plots are images of particular regions or quantities in the simulation. These plots include graphs, contour, histograms, pseudocolor plots, etc. The following are commonly created plots:

- **DAV 1:** Plots of the scalar value potential over time. This requirement primarily draws data from the `3d` file.
- **DAV 2:** Plots of heat flux, turbulence, or the temperature on surfaces over time. This requirement primarily draws data from the `f3d` file.
- **DAV 3:** Plots of the moments of the distributions functions (first order, second order, third order) of the different XGC1 variables: density, kinetic energy, etc. This requirement primarily draws data from the `f3d` file.

Make Videos. Videos show the evolution of the simulation over time. The most common types are field and particle videos. Field videos show the statistical properties of the particles on the mesh. Particle videos show particle evolution, requiring very large amounts of data due to the large number of particles. The plots from DAV 1, DAV 2, and DAV 3 can also be made into videos, but some common



(a) Average PSI velocity in a region in XGC1.

(b) Accumulation of particle impacts to the containment vessel wall.

Figure 2: Example frames from XGC1 analysis videos demonstrating common visualization tasks.

analysis tasks that only make sense when shown as an evolution over time include:

- **DAV 4:** Average vector in a region, as shown in Figure 2(a). This video type primarily uses data stored in the `restart` and `mesh` files.
- **DAV 5:** Rendering particle paths as they progress around the tokamak. This video type primarily uses data stored in the `restart` file.
- **DAV 6:** Detecting and visualizing particles that collide with the tokamak wall, as shown in Figure 2(b). A requirement of this DAV task is the identification of particles that collide with the wall at some point in the simulation. This requires two-passes over the data, one to identify the particles that collide with the wall at any time, and the second to render these identified particles and the collisions with the tokamak wall. After the collision, these particles are removed from the scene. Because of the large size of the particle data, and two passes over all time steps are required, there is no known way to perform this task in situ. Even running the simulation run twice (once to identify particles, and the second time to render identified particles) can be problematic, since the particles are not guaranteed to be reproducible across runs. This video type primarily uses data stored in the `restart` and `mesh` files.
- **DAV 7:** Visualizing the turbulence derived quantity. This video type primarily uses data stored in the `3d`, `mesh`, and `oneddiag` files.

Interactive Visualization and Analysis. Interactive visualization and analysis is accomplished using ADIOS [2] and data staging, where data are streamed from the XGC1 simulation to a data server for visualization. The main interactive visualization task in XGC1 is blob tracking:

- **DAV 8:** Blob tracking involves identifying areas of high energy within the plasma which can form non-linear turbulent eddies. The longevity, size, shape, and composition of these eddies are interesting to researchers, and their visualization gives insight into their 3D structure and perturbation to particle orbits. Blob tracking requires regions of interest to be identified through user interaction, and then the particles composing the blobs in those regions are tracked in subsequent time steps. This task is important because blobs represent areas of high energy and temperature which can damage the wall of the tokamak. Understanding the development and nature of blobs is crucial to the design and operation of tokamaks. The data used in this analysis includes data from the `restart`, `3d`, `mesh`, and `oneddiag` files.

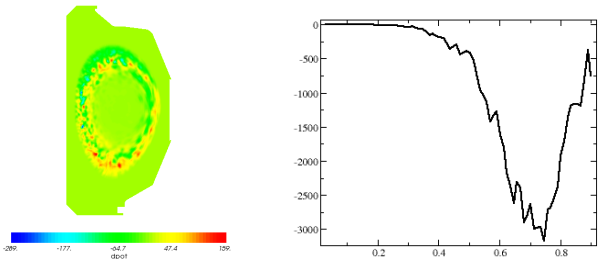
Synthetic Diagnostics. Synthetic diagnostics provide a way to compare simulation and experimental data. Generally, experimental data are not directly comparable to the outputs of simulations, and so a transformation step is often required. Once transformed, experimental data can be used to verify simulation results. These capabilities are currently under development, so no measurable data analysis and visualization task exists yet for this requirement.

4.2. Simulation Monitoring

Simulation monitoring is concerned with real or near-real time reporting of simulation status to the scientists. This monitoring can include tasks such as creating plots of important variables or functions as the simulation progresses, detecting bad simulation states and halting the simulation, and even simulation steering by sending instructions from the monitoring routine back to the simulation.

Simulation Dashboard. A simulation dashboard is an easy to access web page from which scientists can remotely access key information about running simulations, as well as past simulations. For data that cannot be appended to existing plots at each time step, the dashboard must allow a mechanism to explore plots over time. It should enable support for continuing a past simulation run on the same dashboard, and contain links to the storage locations for the data used in each of the visualizations for each run, making retrieval of data related to interesting aspects of a run easy. The dashboard visualization requirements are as follows:

- **DAV 9:** Plotting values on each of the poloidal planes of the simulation for every time step, as shown in Figure 3(a). The number of planes that are plotted are equal to the number of simulated poloidal planes in the tokamak, typically 16, 32, or 64, plus one plot that represents averages of the values of all planes. This requirement primarily draws data from the `3d` and `mesh` files.
- **DAV 10:** Plotting all of the variables contained in the `oneddiag` file for each time step, as shown in Figure 3(b). Typically this produces 150 different plots.
- **DAV 11:** The automatic creation of a video summarizing each variable at the end of the simulation, a video of the average planes from DAV 9, and videos summarizing the slices of the torus.



(a) Example of a slice plot of *potential* at one time step.

(b) Example of a variable plot showing *poloidal flow* over time.

Figure 3: Example images produced by an XGC1 online dashboard during one simulation time step.

4.3. Debugging and Performance Engineering

There are a number of debugging and performance tasks that are desired, or in the works, for XGC1, but, at present, they are not part of the production codebase or analysis and visualization workflows. We therefore have no DAV tasks to report. However, we include a discussion on the major items on the wish list to illustrate directions for future development.

Debugging. Debugging code related to the introduction of new physics or performance enhancements in XGC1 is always challenging. Worse, many problems only occur when running at very large scales.

- *Error Logs* are one method of debugging, and provide a great source of information, though is generally underutilized. The ability for analysis and visualization of these logs could provide useful feedback.
- *Particle Loss* is the loss of particles from the tokamak containment vessel. Recent particle loss has manifested near the simulation boundaries. This information is currently saved to the error log and retrieved after the run is over.

Low Level Monitoring. No low level monitoring exists in the XGC1 code, meaning that the code will not stop itself once the results become invalid. This is an opportunity for improvement. For example, checks to detect when a certain percent of particles have been lost from the simulation (making the results invalid) could be implemented.

Work Division (load balancing). Work division is the process of balancing the distribution of particles to processor ranks in a plane of the simulation. Three possibilities exists for balancing the particles in an XGC1 plane: (1) the toroidal direction, (2) the poloidal direction, or (3) a hybrid combination of the two. For context, the toroidal direction is the long way around the torus, and the poloidal direction is the short way around the torus.

- *Toroidal Load Balancing* is currently being done in production. Experiments indicate this method yields the biggest performance gains.
- *Hybrid Load Balancing* is under experimental development. At this time it is not clear if this type of load balancing would benefit the overall runtime of the simulation. This is due to the fact that poloidal motion is very fast and intuition tells them that it does not end up being a problem. However, further studies into this could be beneficial.
- *Imbalance detection:* XGC1 currently has no mechanisms for detecting when particle imbalance begins to become a detriment to performance, and when a rebalance would be worth the overhead cost. Further work and analysis would prove useful.

Collision Detection. Collision detection is a feature under development, and attempts to balance the simulation by collisions between particles (currently only a single species, but multiple species would be useful). Methods are wanted to visually compare load imbalances by collision versus particle imbalances to answer the question of how these imbalances are different, and how to optimize for both.

5. Conclusion and Future Work

We surveyed a diverse set of people associated with the large-scale fusion simulation code XGC1, gained an understanding of how they work, and cataloged their visualization and analysis requirements for in situ processing. This look at the breadth and depth of in situ requirements for a large-scale simulation code provides valuable insight into the needs of a diverse team. The identified DAV's vary drastically in terms of computational and data resources required, demonstrating a wide breadth of needed in situ flexibility and capability. Finally, we believe the breadth of requirements for XGC1 will be similar for other simulations codes, but that a study such as ours would need to be repeated for these teams to gain an in depth understanding.

As part of our future work, we plan to develop a performance model for each DAV that incorporates computational, time, and resource constraints to predict if it is viable to be performed in situ. This performance model will answer the question of whether or not the integration of in situ visualization and analysis techniques into key areas of a workflow is viable under varying user and system requirements. This predictive capability will allow scientific teams to properly assess, and then appropriately select, the set of visualization tasks that will minimize the impact on the running simulation, while maximizing the extraction of scientific knowledge.

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Award Number 14-017566. The authors thank the XGC1 code team for their participation, and also the anonymous reviewers for their helpful comments.

References

[1] S. Ahern, A. Shoshani, K.-L. Ma, A. Choudhary, T. Critchlow, S. Klasky, V. Pascucci, J. Ahrens, E. Bethel, H. Childs *et al.*, "Scientific discovery at the exascale," *Report from the DOE ASCR 2011 Workshop on Exascale Data Management*, 2011.

[2] Q. Liu, J. Logan, Y. Tian, H. Abbasi, N. Podhorszki, J. Y. Choi, S. Klasky, R. Tchoua, J. Lofstead, R. Oldfield, M. Parashar, N. Samatova, K. Schwan, A. Shoshani, M. Wolf, K. Wu, and W. Yu, "Hello adios: the challenges and lessons of developing leadership class i/o frameworks," *Concurrency and Computation: Practice and Experience*, vol. 26, no. 7, pp. 1453–1473, 2014. [Online]. Available: http://dx.doi.org/10.1002/cpe.3125

[3] The HDF Group, "Hdf5 users guide," https://www.hdfgroup.org/HDF5/doc/UG/, accessed: 6/20/2016.

[4] V. Vishwanath, M. Hereld, and M. E. Papka, "Toward simulation-time data analysis and i/o acceleration on leadership-class systems using glean," in *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*. IEEE, 2011, pp. 9–14.

[5] N. Liu, J. Cope, P. Carns, C. Carothers, R. Ross, G. Grider, A. Crume, and C. Maltzahn, "On the role of burst buffers in leadership-class storage systems," in *Mass Storage Systems and Technologies (MSST), 2012 IEEE 28th Symposium on*. IEEE, 2012, pp. 1–11.

[6] K. Sato, K. Mohror, A. Moody, T. Gamblin, B. R. de Supinski, N. Maruyama, and S. Matsuoka, "A user-level infiniband-based file system and checkpoint strategy for burst buffers," in *Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on*. IEEE, 2014, pp. 21–30.

[7] S. Lakshminarasimhan, N. Shah, S. Ethier, S. Klasky, R. Latham, R. Ross, and N. F. Samatova, "Compressing the incompressible with isabela: In-situ reduction of spatio-temporal data," in *European Conference on Parallel Processing*. Springer, 2011, pp. 366–379.

[8] S. Li, K. Gruchalla, K. Potter, J. Clyne, and H. Childs, "Evaluating the Efficacy of Wavelet Configurations on Turbulent-Flow Data," in *Proceedings of IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, Chicago, IL, Oct. 2015, pp. 81–89.

[9] D. Pugmire, J. Kress, H. Childs, M. Wolf, G. Eisenhauer, R. Churchill, T. Kurc, J. Choi, S. Klasky, K. Wu, A. Sim, and J. Gu, "Visualization and analysis for near-real-time decision making in distributed workflows," in *High Performance Data Analysis and Visualization (HPDAV) 2016 held in conjuction with IPDPS 2016*, May 2016.

[10] H. Zou, Y. Yu, W. Tang, and H. M. Chen, "Improving i/o performance with adaptive data compression for big data applications," in *Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International*. IEEE, 2014, pp. 1228–1237.

[11] J. S. Meredith, S. Ahern, D. Pugmire, and R. Sisneros, "EAVL: the extreme-scale analysis and visualization library," in *Eurographics Symposium on Parallel Graphics and Visualization*. The Eurographics Association, 2012, pp. 21–30.

[12] K. Moreland, U. Ayachit, B. Geveci, and K.-L. Ma, "Dax toolkit: A proposed framework for data analysis and visualization at extreme scale," in *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, Oct 2011, pp. 97–104.

[13] K. Moreland, C. Sewell, W. Usher, L. Lo, J. Meredith, D. Pugmire, J. Kress, H. Schroots, K.-L. Ma, H. Childs, M. Larsen, C.-M. Chen, R. Maynard, and B. Geveci, "VTK-m: Accelerating the Visualization Toolkit for Massively Threaded Architectures," *IEEE Computer Graphics and Applications (CG&A)*, vol. 36, no. 3, pp. 48–58, May/Jun. 2016.

[14] E. P. Duque, D. E. Hiepler, R. Haimes, C. P. Stone, S. E. Gorrell, M. Jones, and R. Spencer, "Epic–an extract plug-in components toolkit for in situ data extracts architecture," in *22nd AIAA Computational Fluid Dynamics Conference*, 2015, p. 3410.

[15] N. Fabian, K. Moreland, D. Thompson, A. C. Bauer, P. Marion, B. Gevecik, M. Rasquin, and K. E. Jansen, "The paraview coprocessing library: A scalable, general purpose in situ visualization library," in *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*. IEEE, 2011, pp. 89–96.

[16] M. Larsen, E. Brugger, H. Childs, J. Eliot, K. Griffin, and C. Harrison, "Strawman: A batch in situ visualization and analysis infrastructure for multi-physics simulation codes," in *Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization*. ACM, 2015, pp. 30–35.

[17] S. G. Parker and C. R. Johnson, "Scirun: a scientific programming environment for computational steering," in *Proceedings of the 1995 ACM/IEEE conference on Supercomputing*. ACM, 1995, p. 52.

[18] C. Chang, S. Ku, P. Diamond, Z. Lin, S. Parker, T. Hahm, and N. Samatova, "Compressed ion temperature gradient turbulence in diverted tokamak edgea)," *Physics of Plasmas (1994-present)*, vol. 16, no. 5, p. 056108, 2009.

[19] R. Blandford, Y.-K. Kim, N. Christ *et al.*, "Challenges for the understanding the quantum universe and the role of computing at the extreme scale," in *ASCR Scientific Grand Challenges Workshop Series, Tech. Rep*, 2008.

[20] W. Washington, "Challenges in climate change science and the role of computing at the extreme scale," in *Proc. of the Workshop on Climate Science*, 2008.

[21] G. Young, D. Dean, M. Savage *et al.*, "Forefront questions in nuclear science and the role of high performance computing," in *Technical report, ASCR Scientific Grand Challenges Workshop Series*, 2009.

[22] W. Tang, D. Keyes, N. Sauthoff, N. Gorelenkov, J. Cary, A. Kritz, S. Zinkle, J. Brooks, R. Betti, W. Mori *et al.*, "Scientific grand challenges: Fusion energy sciences and the role of computing at the extreme scale," in *DoE-SC Peer-reviewed report on major workshop held March*, 2009, pp. 18–20.

[23] R. Rosner, E. Moniz *et al.*, "Science based nuclear energy systems enabled by advanced modeling and simulation at the extreme scale," in *ASCR Scientific Grand Challenges Workshop Series, Tech. Rep*, 2009.

[24] G. Galli, T. Dunning *et al.*, "Discovery in basic energy sciences: The role of computing at the extreme scale," in *ASCR Scientific Grand Challenges Workshop Series, Tech. Rep*, 2009.

[25] R. Stevens, M. Ellisman *et al.*, "Opportunities in biology at the extreme scale of computing," in *ASCR Scientific Grand Challenges Workshop Series, Tech. Rep*, 2009.

[26] A. Bishop, P. Messina *et al.*, "Scientific grand challenges in national security: The role of computing at the extreme scale," in *ASCR Scientific Grand Challenges Workshop Series, Tech. Rep*, 2009.

[27] 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 *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*. IEEE, 2012, pp. 1–9.

[28] D. Ellsworth, B. Green, C. Henze, P. Moran, and T. Sandstrom, "Concurrent visualization in a production supercomputing environment," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 12, no. 5, pp. 997–1004, 2006.

[29] P. Malakar, V. Vishwanath, T. Munson, C. Knight, M. Hereld, S. Leyffer, and M. E. Papka, "Optimal scheduling of in-situ analysis for large-scale scientific simulations," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, 2015, p. 52.

[30] M. Slawinska, M. Clark, M. Wolf, T. Bode, H. Zou, P. Laguna, J. Logan, M. Kinsey, and S. Klasky, "A maya use case: adaptable scientific workflows with adios for general relativistic astrophysics," in *Proceedings of the Conference on Extreme Science and Engineering Discovery Environment: Gateway to Discovery*. ACM, 2013, p. 54.