

Building a Transdisciplinary, Exascale-Capable Workforce for Geospace Science

Authors: C. Bard¹, K. Sorathia², G.L. Delzanno³, J. Dorelli¹, J. Lyon⁴, S. Ukhorskiy², M. Wiltberger⁵

Key Points:

- First-principles, self-consistent geospace modeling will require at least exascale-level computing capabilities; however, the technical skills necessary to develop such simulation codes are not taught as part of Heliophysics training/PhD programs.
- Developing exascale Heliophysics codes will require transdisciplinary collaborations between physicists, computer scientists, software engineers, data scientists, and applied mathematics. Such teams must be persistent and formed around specific skills, not specific problems.
- We must have stable, positive long-term career outcomes for Heliophysics scientific developers in order to retain knowledge of and promote future innovation in scientific algorithm development and advanced computing techniques. One such way to do this is by funding long-term scientific programming groups, akin to Heliophysics DRIVE Centers.
- We must have a balance between promoting innovation via funding short-term closed-source development and allowing the open-source community to benefit from and build on the newest scientific modeling techniques.

¹ NASA Goddard Space Flight Center

² Johns Hopkins Applied Physics Laboratory

³ Los Alamos National Laboratory

⁴ Dartmouth University

⁵ National Center for Atmospheric Research

A Need for a Exascale-Capable Community

One of the great challenges for Heliophysics over the next decade is the necessity of large-scale global simulations that encompass the full magnetospheric “system of systems” without separating the full problem into models of individual particle populations and domains. As detailed in the whitepaper by Sorathia et al.⁶, our current “model of models” approach misses key physical effects in the transition regions between models; thus, a global simulation is needed which can simultaneously provide complete physics and high resolution throughout the entire magnetospheric domain. Indeed, several approaches to go “beyond MHD” are being developed, including extended MHD, multi-fluid, hybrid, Vlasov, spectral, and implicit solvers for both plasma and neutral gases. However, while these models promise great scientific advances, severe computational limitations require tradeoffs such as limited resolution, simplified approximations, or reduced dimensionality.

Although these detailed simulations have not been computationally feasible in the past, we are now in the era of exascale supercomputing: Frontier at Oak Ridge’s Leadership Computing Facility now exceeds 10^{18} floating point operations per second (FLOPS). **We must take advantage of exascale computing (and beyond) in order to enable cross-scale and self-consistent magnetospheric modeling.** In addition to overcoming scientific challenges, **this task will require that Heliophysics overcome institutional challenges to create and sustain a cross- and trans-disciplinary workforce spanning applied mathematics, physics, computer science, data science, and software engineering.** These are skills in very high demand, and Heliophysics will have to compete with industries that can offer high salaries, long-term stability, and work-life balance.

A Team of Teams

Fully harnessing exascale platforms to produce exaflop performance is an enormous challenge. It requires intricate knowledge of computer science and parallel algorithms, especially with regards to concurrency, memory handling, and asynchronous task scheduling across multiple computational platforms and architectures. Indeed, Heldens et al. [2020]⁷ state that “two major concerns [in exascale computing research are] the lack of suitable programming tools and the growing gap between processor performance and data bandwidth (i.e., memory, storage, networks).” The only way to fulfill the breadth of expertise required is to form large and persistent teams. However, **Heliophysics (and other subdisciplines of physics) does not provide the specialized training to solve these issues and write code that scales well to exaflop performance.**

⁶ “First-Principles Geospace Modeling in the Next Decade”, K. Sorathia et al., 2022 Heliophysics Decadal White Paper

⁷ S. Heldens, P. Hijma, B. Van Werkhoven, J. Massen, A.S.Z. Belloum, R.V. van Nieuwpoort, “The Landscape of Exascale Research: A Data-Driven Literature Analysis”, *ACM Comput. Surv.*, 53, 2020; <https://doi.org/10.1145/3372390>

As a result, the most advanced scientific codes today are split between physics (solvers for, e.g., plasma, radiation, and/or gravitational effects) and computer science (grid-adaptive, concurrent, platform-agnostic algorithms). These codes are a product of large, multi-institution collaborations between physicists and computer scientists. Some currently under development include Athena++/Parthenon^{8,9}, Enzo-E/Cello¹⁰, FLASH-X/AMRex¹¹, CASTRO/AMRex¹², and WDMapp¹³, among many others. This is a marked difference from what has been done historically. Many scientific codes (including the forerunners of the above-mentioned collaborations) have been developed through funding from various grants, being built by small teams of tenured physics researchers and/or graduate students and postdocs. **There must be dedicated and sustainable effort and funding to assemble teams of physicists, computer scientists, software engineers, data scientists, and applied mathematicians to work on exascale codes full time with stable jobs, rather than piecemeal funding of small grants and groups in short-term projects.**

The piecemeal, short-term nature of scientific code development hinders long-term success: developers with the most relevant recent experience in designing, debugging, and innovating scientific code algorithms often do not stay in the field after their graduate studies or short-term postdoc. This not only requires investing and reinvesting time to train new workers, but also leads to a continual erosion of community knowledge pertaining to computational code development. **We must have stable, positive long-term career outcomes for scientific developers in order to retain knowledge of and promote future innovation in scientific algorithm development and advanced computing techniques.**

Grant proposal funding is often focused around individual problems and specialized solutions. **For long-term stability and flexibility, teams should be built around skills and not problems.** Such teams would not be formed to build a specific model, but instead would focus on exploring new approaches to solve scientific and computational problems, being flexible in determining the best methodology. These teams must not only be able to prototype new algorithms quickly, but also simultaneously work on methods to assimilate data and deploy on exascale resources. This is a “team of teams” model wherein subgroups from different disciplines are able to continuously collaborate and iterate on new models and algorithms. Long-term, stable teams also greatly benefit the broader Heliophysics community: having a consistent source of funding aligns team incentives with open-source and open-science goals¹⁴ without having to hoard knowledge to ensure survival in the next round of proposals.

⁸ Stone, J.M., Tomida, K., White, C.J., et al., 2020, *Astrophys. J., Suppl. Ser.*, 249, 4.; doi:10.3847/1538-4365/ab929b

⁹ <https://github.com/lanl/parthenon>

¹⁰ <https://github.com/enzo-project/enzo-e>

¹¹ <https://flash-x.org/>

¹² <https://amrex-astro.github.io/Castro/>

¹³ <https://sites.google.com/pppl.gov/ecpwdmapp>

¹⁴ See the NASA TOPS mission (<https://science.nasa.gov/open-science/transform-to-open-science>) for discussion on open science.

Challenges in Creating an Exascale Community

There are several challenges that must be overcome in order to have a healthy, skilled, exascale-capable developer base in Heliophysics:

- *Recruitment of necessary skills*: Computer programmers are increasingly in demand in our hyper-technological world. We will not be able to keep skilled people in our field if they are to work long hours for apprentice wages and no job stability.
- *Early-Career retention*: Graduate students who primarily develop geospace models have generally worse career outcomes than the people who are "model early adopters". Time spent writing code is time not spent writing papers, which is a severe issue when hiring and promotion decisions for early career scientists is almost entirely informed by first-author publications.
- *Early-Career stability*: Similarly, applicants to most "soft-money" positions are judged by first-author publications. It is difficult to continue program development and refinement while simultaneously publishing at the level required to continue working in Heliophysics.
- *Support for software engineers and/or scientific programmers*: There must be consistent funding and long-term stability for developers to develop and maintain advanced simulation codes. This includes proper recognition of importance and effort, especially if their primary job does not involve writing scientific papers.
- *Support for community-usable codes*: There is a significant increase in the time/effort required to create a research code that is generically useful for the community over one that is usable for an individual researcher. Additionally, it takes time and effort to maintain codes for a dedicated user-base, including optimizations and quality-of-life improvements. Funding models typically do not support this extra effort.
- *Utility to the community/open-source*: Although NASA will soon require open-source code release for projects funded with their grants (starting in 2024), competition for this funding encourages closed-source codes. Since future grant success hinges on publishing papers under current grants, there is a competitive advantage to limiting access to scientific source codes and their model outputs. However, this does not promote innovation nor allow the creation of a community to use and build on prior successes. On the other hand, having open code development would promote innovation and also exacerbate the divide between "developers" and "early adopters" (as mentioned in point 2).

Recommendations

We provide here several recommendations with the intent to incite discussion about how to unify the disparate realms of Heliophysics scientific programmers into a sustainable, stable, and innovative community:

Create Scientific Programmer Positions: In addition to having permanent positions dedicated to scientific research, i.e., publication of papers, **we recommend that institutions create stable, long-term positions dedicated to scientific coding and algorithm development**. Hiring and promotion for such positions would weigh contributions to useful scientific codes equally as much as (or more heavily than) publishing scientific results.

Scientific Code “DRIVE Centers”: One of the key recommendations from the previous Heliophysics Decadal was the creation of heliophysics science centers, with long-term funding, to tackle major science problems by forming interdisciplinary teams. These DRIVE Centers unified observers, theorists, modelers, and computer scientists across disciplinary boundaries. For scientific code development, we need something similar. However, rather than interdisciplinary teams, **we need to create long-lived *transdisciplinary* teams of physicists, computer scientists, applied mathematicians, data scientists, and software engineers** to tackle difficult problems in Heliophysics exascale development. Such teams would not only have scientists who prototype physical models but also engineers and developers who deploy on exascale clusters and help with algorithm optimizations. With a source of steady funding, the code produced by these teams can be open-source throughout the full development cycle without hindering long-term career outcomes for team members.

Bifurcated Funding Stream for Innovation vs. Open Source: As mentioned above, current funding mechanisms produce a tension between scientific code innovation and community utility. In addition to funding “DRIVE centers for coding” (as mentioned above), we propose a bifurcated funding stream for scientific code development: 1) **Dedicated, persistent funding for the continued development and maintenance of multiple open source exascale codes**, and 2) **funding for private development and prototyping of codes which will be openly released shortly after the grant period ends** (in order to spur and drive innovation). By combining these two avenues, we can encourage short-term bursts of innovation which can be continuously folded and adapted into stable, long-term community codes.

Summary

Heliophysics over the last few decades has enjoyed much success in developing both global magnetospheric models and detailed physical simulations of local phenomena. However, the key challenge in the next decade is to have our cake and resolve it too: global simulations without compromising on modeling local effects and feedback. These codes will not only require exascale (and beyond) computing resources, but also the creation and sustenance of a long-term scientific developer community in Heliophysics. **The challenges of exascale computing require that our community be *trans-disciplinary*:** not only physicists, but also computer scientists, software engineers, data scientists, and applied mathematicians. We need people to prototype new codes *and* optimize them for deployment on exascale resources.

Recruiting and retaining such a workforce is challenging, especially since the skills required for exascale computing are in high demand in our hyper-technological world. Unfortunately, career outcomes in science for developers are often worse than for “first adopters” who are able to spend more time writing papers instead of code. This is a result of predominantly valuing first-author publications in hiring and promotion decisions. **We must value contributions to scientific codes equally as much as publications.** There must be an intent to keep the necessary skills within our community. Otherwise, developers will leave the community for jobs with better pay and life-work balance, leading to the degradation of both present-day knowledge and future innovation. One way to keep skills in our community is to provide long-term, stable

funding. **We recommend the creation and long-term funding of transdisciplinary teams, akin to scientific coding DRIVE Centers, that can explore new approaches for geospace simulations.**

Finally, a robust and thriving sustainable exascale community requires both innovation and openness. However, current funding models which promote innovation also encourage closed-source codes in order to maintain a competitive advantage for future grant proposals. We need a structure that brings new algorithms to open release, allowing them to be more easily incorporated in community codes, while also recognizing and valuing the time and effort required to create these breakthroughs. **We propose a bifurcated funding model in which there is dedicated funding for both open-source exascale codes and short-term closed-source development.** After the short-term development period, the new advances will be released and then incorporated into existing community codes.

Heliophysics is at the precipice of a new era in making fundamental discoveries with exascale computing. But, **it is not sufficient to have computers; we must also have people** who are able to utilize these new resources.