siam news

SIAM News Blog SOFTWARE AND PROGRAMMING

Supporting Computational Science and Engineering: The Creation of Widely Used Software in Industrial and Applied Mathematics

October 14, 2025

By Daniel Arndt, Wolfgang Bangerth, Bruno Blais, Marc Fehling, Rene Gassmöller, Timo Heister, Luca Heltai, Martin Kronbichler, Matthias Maier, Peter Munch, Bruno Turcksin, and David Wells

Industrial and applied mathematicians, engineers, computational scientists, physicists, and other scientific researchers routinely use components of the vast universe of open source mathematical software. In many cases, they directly utilize popular open source packages. In other cases, these uses are more subtle; even closed source, commercial products such as MATLAB, Ansys, Ansys Fluent, Abaqus, and in-house codes in companies like Boeing or Airbus build important parts of their functionality on software that either is open source or derived from open source projects that originate within the applied math community.

In recent years, SIAM has recognized the importance of mathematical software by awarding the biennial SIAM/ACM Prize in Computational Science and Engineering to the authors of widely used software packages. In 2015, the developers of PETSc—a package that provides parallel linear and nonlinear solvers for dense and sparse problems, along with other functionalities—received the prize, while the principals of the differential equations solver package SUNDIALS were honored in 2023. This year, the 2025 award recognized the authors of this article for the creation of deal.II: a comprehensive finite element package.

Given these recent acknowledgments, it is perhaps a good time to talk about the creation of such software. Indeed, the investment that these packages' developers make is fundamentally different from the investment into the many software projects that are written within individual research groups or collaborations. In the following, we will frequently mention statistics that pertain to deal.II because it is the project with which we are most familiar, but many other mathematical software projects are of similar size and can serve the same purpose.

Large Mathematical Software Packages Are Not Like Home-grown Scientific Software

At the 2025 SIAM Conference on Computational Science and Engineering, which took place in Fort Worth, Texas, this past March, the principal authors of the deal.II project received the 2025 SIAM/ACM Prize in Computational Science and Engineering for their creation of a highly impactful library that supports finite element calculations. From left to right: Daniel Arndt of Oak Ridge National Laboratory, Wolfgang Bangerth of Colorado State University, Timo Heister of Clemson University, Guido Kanschat of Heidelberg University in Germany, Martin Kronbichler of Ruhr-Universität Bochum in Germany, Matthias Maier of Texas A&M University, Peter Munch of the Technical University of Berlin, Bruno Turcksin of Oak Ridge National Laboratory, and David Wells of the University of North Carolina at Chapel Hill. Not pictured are Bruno Blais of Polytechnique Montréal in Canada, Marc Fehling of Charles University in the Czech Republic, Rene Gassmoeller of the GEOMAR Helmholtz Centre for Ocean Research Kiel in Germany, Luca Heltai of the University of Pisa in Italy, and consultant Jean-Paul Pelteret. SIAM photo.

Let's begin with an observation that is perhaps not obvious: Packages such as PETSc, SUNDIALS, and deal.II are not just super-sized graduate projects that later became the basis of a research group or collaboration's efforts. Excluding its test suite, deal.II has roughly 800,000 lines of code; PETSc and SUNDIALS have about the same, and Trilinos—another library for linear and nonlinear solvers—has over three million. The SLOCCount program estimates that the creation of deal.II—excluding the test suite and tutorials—required 122 person-years of work, which in today's salaries and overhead would cost upwards of $50 million. More than 400 people have contributed throughout its 27-year history, and every day sees about five new commits. We know of at least 2,700 publications that detail results obtained with the help of deal.II in fields from aerodynamics to zoology; the true number is likely much larger.

The aforementioned software packages have all existed for over 25 years. At their scale of size, time, and contributor counts, very different principles of software development apply than for projects with a few tens of thousands of lines that are developed over several years by a handful of co-located people. For these types of large enterprises, no single individual can be truly knowledgeable about even a portion of the entire code base, and projects must pay great attention to issues such as modularity, adequate documentation, and automated testing. Software that is used widely in both academia and commercial products is also held to a different level of rigor; it must compute correct answers, remain compatible with a broad range of computing platforms, contain useful documentation, and maintain a high degree of backwards compatibility. The introduction of bugs during continued development is simply not acceptable.

Consequently, most large mathematical software packages have invested significant time into infrastructure that ensures patches do not break existing functionality. For example, deal.II runs a test suite with more than 17,000 tests daily on multiple machines. A typical workstation takes about 48 processor hours to complete the entire test suite. These tests also automatically run in multiple configurations and across various operating systems before any changes can be merged into the library via GitHub pull requests, ensuring correctness and compatibility. Moreover, every patch—even if it just changes a comma in a comment—is reviewed by at least one principal developer; the deal.II principal developers are not exempt from this requirement and are not allowed to merge their own patches without approval by their peers.

Mathematical software is a representation of the state of the art in the implementation of mathematical algorithms. It is also often a tool for teaching this state of the art, and much work goes into the education of current and future users. For example, deal.II currently has 88 tutorial programs, 25 "code gallery" codes contributed by users, and more than 80 YouTube-hosted video lectures that offer instruction on the use of deal.II, finite element methods, and computational science and engineering in general. Collectively, these teaching materials likely correspond to another 10 person-years of work.

Large-scale Software Is a Career Commitment

Given the amount of time that goes into writing the software itself, producing extensive test suites and the infrastructure to run them, reviewing patches, and providing documentation, it is perhaps surprising that few people are explicitly paid to create these packages. For example, the 13 principal developers of deal.II have jointly received less than $5 million in grants for the library's development over the past 25 years, roughly half of which was for salaries — a fairly small fraction of the estimated cost of creating deal.II. Instead, much of the work is either co-financed by academic appointments or a side product of other funded projects.

Moreover, the authors of large scientific software projects put years of work into something that does not fit the typical categories of academic productivity — at the cost of peer-reviewed journal publications. Most academic departments do not understand the level of commitment that is necessary to produce high-quality, widely usable software. In promotion or tenure meetings, many individuals who write scientific software have likely heard statements such as, "We all write software, what's the big deal?" But as explored above, writing broad-reaching software is a big deal. It is also worth considering the pervasive impact of this work; these software packages enable whole communities to be much more productive than they would otherwise be if everyone had to write their own software — much like how a subset of astronomers construct telescopes for a greater audience, and specialized physicists build particle accelerators for everyone. In contrast to the infrastructure for physics experiments, mathematical software is not usually tied to one community. Like many other packages, deal.II is used in thousands of publications that span nearly every imaginable discipline in both academia and industry. Such software transcends the home disciplines of its authors and—in the context of published academic work—has a broad impact on science.

Wolfgang Bangerth of Colorado State University delivered a presentation about the deal.II project during the 2025 SIAM Conference on Computational Science and Engineering, which was held in Fort Worth, Texas, this past March. As the principal authors of deal.II, Bangerth and his colleagues received the 2025 SIAM/ACM Prize in Computational Science and Engineering. SIAM photo.

Community Matters

Most large mathematical software packages are created by *communities*, not individuals. Hundreds of people—from undergraduate students to retirees on all seven continents—have contributed to deal.II. So it indeed takes a village, but this village is constantly changing. While contributors often start to work on open source software as graduate students, many of them stop when they graduate and leave academia. Others continue until they are young faculty members but eventually shift their focus when they begin to build their own research groups and spend more time on managerial activities. In any case, continued software development requires both a core of dedicated, long-term active principals and a continuous pipeline of new talent with the time to collaborate. This pipeline does not make itself; it involves mentoring and encouraging individuals, teaching in other countries, and supporting those who show promise [1]. It also necessitates that one relinquish control and give others the opportunity to prove themselves as project maintainers. As with code, infrastructure, and documentation, developers spend much time building and maintaining the pipeline of current and future contributors to sustain projects.

Summary and Key Takeaways

Open source software plays a crucial role in industrial and applied mathematics by supporting a wide range of academic research and commercial applications. Packages like PETSc, SUNDIALS, and deal.II have grown into large, mature projects developed by hundreds of contributors over several decades, far exceeding typical software products in scale and complexity. These packages require extensive testing, detailed documentation, and rigorous review processes to maintain correctness, accuracy, quality, and compatibility. Their sustainability relies heavily on active, collaborative communities; ongoing mentoring; and new contributors to ensure continuity and long-term success. Despite the enormous collective effort—often hundreds of person-years—individuals typically receive limited financial support or academic recognition. Nevertheless, open source mathematical software significantly boosts scientific productivity across disciplines and promotes open science that benefits academia, industry, and broader society. We deeply appreciate the SIAM/ACM Prize in Computational Science and Engineering for recognizing the importance of these software packages and the communities that create them.

Wolfgang Bangerth delivered a prize talk about deal.II in response to his team's receipt of the 2025 SIAM/ACM Prize in Computational Science and Engineering at the 2025 SIAM Conference on Computational Science and Engineering, which took place earlier this year in Fort Worth, Texas.

References
[1] Bangerth, W. (2019, June 27). Leading a scientific software project: It's all personal. *Better Scientific Software Blog*. Retrieved from https://bssw.io/blog_posts/leading-a-scientific-software-project-it-s-all-personal.

CSE25

About the Authors

Daniel Arndt

Computational scientist, Oak Ridge National Laboratory

Daniel Arndt is a computational scientist in the Scalable Algorithms and Coupled Physics group at Oak Ridge National Laboratory.

Wolfgang Bangerth

Professor, Colorado State University

Wolfgang Bangerth is a professor of mathematics at Colorado State University. In 1997, he founded and is now one of the principal developers of the deal.II project that provides finite element functionality from laptops to supercomputers.

Bruno Blais

Associate professor, Polytechnique Montréal

Bruno Blais is an associate professor in the Department of Chemical Engineering at Polytechnique Montréal in Canada.

Marc Fehling

Postdoctoral fellow, Charles University

Marc Fehling is postdoctoral fellow within the Faculty of Mathematics and Physics at Charles University in the Czech Republic.

Rene Gassmöller

Staff scientist, GEOMAR Helmholtz Centre for Ocean Research Kiel

Rene Gassmöller is a staff scientist at the GEOMAR Helmholtz Centre for Ocean Research Kiel in Germany.

Timo Heister

Professor, Clemson University

Timo Heister is a professor in the School of Mathematical and Statistical Sciences at Clemson University.

Luca Heltai

Associate professor, University of Pisa

Luca Heltai is an associate professor of numerical analysis in the Department of Mathematics at the University of Pisa in Italy.

Martin Kronbichler

Professor, Ruhr-Universität Bochum

Martin Kronbichler is a professor of applied numerics at Ruhr-Universität Bochum in Germany.

Matthias Maier

Associate professor, Texas A&M University

Matthias Maier is an associate professor in the Department of Mathematics at Texas A&M University.

Peter Munch

Postdoctoral researcher, Technical University of Berlin

Peter Munch is a postdoctoral researcher in the Chair of Numerical Methods for Partial Differential Equations within the Institute of Mathematics at the Technical University of Berlin in Germany.

Bruno Turcksin

Computational scientist, Oak Ridge National Laboratory

Bruno Turcksin is a computational scientist in the Computational Sciences and Engineering Division at Oak Ridge National Laboratory.

David Wells

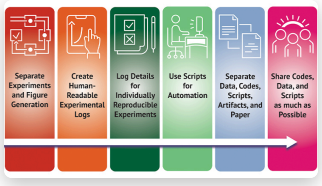
Research scientist, University of North Carolina at Chapel Hill

David Wells is a research scientist at the University of North Carolina at Chapel Hill.

Related Reading

Taming the Chaos of Computational Experiments

September 2, 2025



Forward-looking Panel at CSE25 Contemplates Future Directions of Computational Science and Engineering

May 1, 2025

Introducing the Consortium for the Advancement of Scientific Software

March 4, 2025