Computational Science

Introduction: Computational science is a multidisciplinary science that integrates applied mathematics, computer science, and domain science to simulate and analyze complicated real-world systems. Computational science has been referred to as the "third pillar" of science, in addition to theory and experiment. Through building mathematical models of natural phenomena and executing numerical simulations using computers, computational science empowers scientists to study problems that are impossible experimentally or analytically. For instance, computer techniques have enabled the design of new drugs, the creation of new industrial products, and precise weather and climate forecasting 2. Briefly, computational science employs mathematics and computers "to drive progress in science and engineering" and facilitates most contemporary technologies.

Theory: Computational science is the basis of numerical algorithms and mathematical modeling at its foundation. Researchers start with deriving equations (differential equations or linear algebra representations, for instance) that model a system from first-principles or empirical laws. The formulation of the model is followed by the selection of suitable numerical methods for approximating the solution (finite element or finite difference methods for partial differential equations, for instance, or Monte Carlo for stochastic processes, etc.). These methods must be developed with algorithmic sensibility – a sensitivity to computational complexity, numerical stability, and convergence. Implementations are often tuned for performance on contemporary hardware (through parallel algorithms, vectorization, and high-performance libraries). As one survey notes, computational science "brings together computer simulation, scientific visualization, mathematical modeling, computer programming, data structures, networking, [and] high-performance computing" to address scientific and engineering problems. An end-to-end workflow typically includes model verification and validation (against known solutions or experimental data) and full analysis and visualization of results. Practicing computational scientists need to trade off model fidelity against computational cost, typically iterating between

algorithmic optimization, model refinement, and optimization of the code for performance.

Subfields: Computational science is very broad. It includes classical fields such as computational physics, computational chemistry, and computational biology, and newer ones (e.g. computational social science). Computational physics, for example, examines numerical methods for physical problems – basically "study and application of numerical analysis to solve problems in physics". Very closely related, computational chemistry utilizes simulation (often quantum and molecular dynamics) to model chemical systems and reactions. Computational biology is the use of algorithms on biological data and models; a subdiscipline of biology that includes "the use of computers and computer science to understand and model the structures and processes of life," using computational methods to simulate biological systems and analyze large-scale data. Some other subfields are computational engineering (computational fluid dynamics, structural analysis, e.g.), climate and Earth system modeling (atmosphere, ocean, land processes simulation), computational finance (markets modeling), etc. Current overview stresses that today's computational science encompasses fields "such as computational physics, computational biology, computational chemistry, [and] computational social science", etc. In all of these disciplines, researchers construct domain-specific models (e.g. Navier–Stokes equations for fluid dynamics, Schrödinger equations for quantum chemistry, agent-based models for social systems) and solve them computationally.

Case Study – Climate Simulation: A prime example of computational science is in climate modeling. Climate science today employs large-scale numerical models of the Earth's atmosphere, oceans, land surface, and ice to make predictions about future climate and extreme events. As a case in point, a recent project created a global cloud-resolving climate model (SCREAM/E3SM) and executed it on the Frontier exascale supercomputer.

Here, the physics and thermodynamics of fluid flows (the Navier–Stokes equations with radiation and convection models) are discretized on a very fine global grid. With 1.1 exaflops of computing power from Frontier, scientists could simulate one year of global climate in a single day. This record-breaking world speed was realized by the parallelization of the code

over millions of cores and explicit simulation of small-scale cloudiness, necessary for reliable precipitation prediction. These simulations are computationally extremely intensive: distributed workflows analyze the output (temperature, humidity, winds, etc.) to extract trends (e.g. regional drought risk). Climate modeling is the quintessential example of the computational science cycle: a high-fidelity physical model was developed (Earth system equations), implemented in HPC form (optimized for Frontier's architecture), and executed as a "numerical experiment." The outcomes (e.g. better precipitation predictions) illustrate how computational approaches can uncover system behavior that would be impossible to measure directly.

Figure: The Frontier exascale supercomputer at the Oak Ridge National Laboratory, where huge climate simulations were run (image: ORNL, 2023).

Design/Methodology: The standard computational science project follows a serial methodology. Experts in the domain first create a mathematical model of the system (e.g. conservation laws or statistical models). The numerical analysts then select discretization methods and solvers for the model equations. For instance, partial differential equations may be solved using finite difference or finite element methods, and large linear systems may be solved using iterative solvers (e.g. Krylov subspace methods). The model is subsequently coded (frequently in low-level languages such as Fortran or C++, maybe with the support of domain-specific frameworks or libraries). The implementation has to be parallelized for high-performance computing (MPI, OpenMP, GPU programming, etc.). The model is calibrated/validated after coding by comparing outputs to known solutions or experiments. The simulation is ultimately run on high-performance computers, maybe with weeks of runtime and terabytes of output. Then, the results are interpreted and visualized using scientific visualization software tools by analysts in order to understand the results, gain insights, and also calibrate the model. Software engineering principles are essential throughout this flow: version control, testing, and modularity keep the complicated code correct and maintainable. In conclusion, computational science methodology fills the gap between a theoretical model and an executable simulation with mathematics, algorithms, and software engineering being used at every stage.

Relationships to Other Domains: Computational science has strong relationships with several adjacent fields. It relies heavily on applied mathematics (for modeling and numerical analysis) and on computer science (for data structures, algorithms, and high-performance computing). Computational scientists need to be good software engineers in practice as well, as efficient, correct scientific code is crucial. There is also increasing synergy with data science and AI. For instance, machine learning is being applied to simulations (surrogate modeling, uncertainty quantification, and data assimilation). The SIAM report, in fact, identifies new core competencies such as "digital twins" (real-time, data-driven computational replicas of physical systems) and "artificial intelligence–driven modeling" as areas of opportunity. Furthermore, computational science issues have a tendency to yield very large data sets (simulation outputs, experimental data) whose analysis can be leveraged using data-science techniques. That is, both computational science and data science utilize tools (statistics, optimization) with different focus: the former on physics-based modeling and the latter on empirical data. Likewise, there is intersection with operations research/optimization (particularly in engineering design), machine learning (for model replication), and domain sciences (as a method of inquiry in biology, chemistry, and social sciences). These interdisciplinarity relations have the effect that a breakthrough in one field (e.g., AI or software engineering) can have a profound impact on computational science, and vice versa.

Challenges and Future Trends: Computational science holds enormous promise, yet faces serious challenges. Perhaps the biggest concern is hardware limits: traditional gains from CPU speedup have stalled ("Moore's Law has run its course"), so future performance must come from parallelism and specialization. As considered by the SIAM Task Force, future systems will be heterogeneous architectures with GPUs, FPGAs, and even quantum accelerators. Porting software to these platforms is one of the biggest challenges. Shortage of workforce is another: computational science is multidisciplinary, and more experts who can integrate science, mathematics, and computing are needed. Reproducibility and software sustainability for large codes are also a top priority. Two emerging trends in the future stand out. First, coupling with AI and data-based techniques: machine learning can potentially speed up simulations by orders of magnitude (e.g., through learned subgrid models or accelerated surrogate predictions). Second, the development of digital twins and real-time modeling, where real-time data

streams update computational models for decision-making in real time. To take complete advantage of these trends, the community will also need to resolve underlying challenges in uncertainty quantification, multiscale modeling, and algorithms. In short, computational science stands at an "inflection point": computation will demand new training and algorithms as well as sustained investment in high-end computing hardware.

Key Players: there are numerous pioneers who have defined the field. Early pioneers include Alan Turing and John von Neumann, who established the foundations of computer simulation. Nicholas Metropolis (one of the creators of the Monte Carlo method) and Stan Ulam were prominent figures in computational physics. In more recent times, Berni J. Alder is referred to as "one of the pioneers of molecular dynamics simulations", revolutionizing statistical mechanics and chemistry with atomistic computation. Another contemporary leader is Jack Dongarra, a winner of the 2021 ACM Turing Award "for pioneering contributions to numerical algorithms and libraries that enabled high-performance computational software". His libraries (e.g., LAPACK, ScaLAPACK, MPI implementations) have supported much of computational science. Other key players are simulation icons like Margaret Hamilton (software engineering) and domain leaders who promoted computation (e.g., David Keyes in HPC math, Ian Foster in grid/cloud computing). These and numerous others have created the algorithms and software facilitating computational science.

Journals and Conferences: Some of the key journals in this area are the SIAM Journal on Scientific Computing (SISC), which publishes papers on numerical methods and algorithms for scientific computing, and the Journal of Computational Physics (JCP), an Elsevier computational physics journal. Some other important journals are Journal of Scientific Computing (Springer), Computational Science & Discovery (IOP), and International Journal of High Performance Computing Applications. In conferences, ACM/IEEE Supercomputing conference (SC) is the premier yearly conference for simulation and HPC, and SIAM supports a Conference on Computational Science and Engineering (CSE). IEEE/ACM Parallel & Distributed Processing Symposium (IPDPS) and ACM International Conference on Supercomputing (ICS) are also top venues. There are a number of domain-specific symposia (e.g. for fluid dynamics, climate, biology) that include computational sessions. These conferences and journals

publish the newest computational techniques and applications throughout science and engineering.

Bibliography:

• Ceperley, D. M., & Libby, S. B. (2021). Berni Julian Alder, theoretical physicist and molecular dynamics inventor, 1925–2020. Proceedings of the National Academy of Sciences, 118(1), e2024252118.

• Singer, N. (2023, April 6). Cloud-resolving climate model meets world's fastest supercomputer. Sandia National Laboratories Lab News.

• Society for Industrial and Applied Mathematics. (2024). The Future of Computational Science: SIAM Task Force Report.

• Society for Industrial and Applied Mathematics. (n.d.). What Are Applied Mathematics, Computational Science, and Data Science? (SIAM Career Resources).

• Searls, D. B. (n.d.). Computational biology. Encyclopædia Britannica.

What Are Applied Mathematics, Computational Science and Data Science | SIAM

https://www.siam.org/programs-initiatives/professional-development/career-resources/what-are-applied-mathematicscomputational-science-and-data-science/

siam.org

https://www.siam.org/media/ofaijdru/siam-report-on-the-future-of-computational-science.pdf

assets.press.princeton.edu

https://assets.press.princeton.edu/chapters/s1_10291.pdf

(PDF) Simulation and Modeling as the Essence of Computational Science https://www.researchgate.net/publication/326265466_Simulation_and_Modeling_as_the_Essence_of_Computational_Science

Computational physics - Wikipedia https://en.wikipedia.org/wiki/Computational_physics

Computational biology | Algorithms, Data Analysis & Modeling | Britannica https://www.britannica.com/science/computational-biology

Cloud-resolving climate model meets world's fastest supercomputer – LabNews https://www.sandia.gov/labnews/2023/04/06/cloud-resolving-climate-model-meets-worlds-fastest-supercomputer/

Berni Julian Alder, theoretical physicist and inventor of molecular dynamics, 1925–2020 - PMC https://pmc.ncbi.nlm.nih.gov/articles/PMC7980442/

Jack Dongarra - Wikipedia https://en.wikipedia.org/wiki/Jack_Dongarra

SIAM Journal on Scientific Computing | SIAM https://www.siam.org/publications/siam-journals/siam-journal-on-scientific-computing/

Journal of Computational Physics - Wikipedia https://en.wikipedia.org/wiki/Journal_of_Computational_Physics