March 2014

Applied Mathematics Research for Exascale Computing

Exascale Mathematics Working Group

Jack Dongarra	co-chair, ORNL
Jeffrey Hittinger	co-chair, LLNL
John Bell	LBNL
Luis Chacon	LANL
Robert Falgout	LLNL
Michael Heroux	SNL
Paul Hovland	ANL
Esmond Ng	LBNL
Clayton Webster	ORNL
Stefan Wild	ANL

DOE/ASCR Point of Contact
Karen Pao

Sponsored by:

U.S. Department of Energy

Office of Science

Advanced Scientific Computing Research Program

Availability of This Report
This report is available, at no cost, at http://www.osti.gov/bridge. It is also available on paper to the U.S. Department of Energy and its contractors, for a processing fee, from:

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
phone (865) 576-8401
fax (865) 576-5728
reports@adonis.osti.gov

Auspices Statement
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL report number LLNL-TR-651000.

Applied Mathematics Research for Exascale Computing

Sponsored by
U.S. Department of Energy
Office of Science
Advanced Scientific Computing Research Program

Exascale Mathematics Working Group
Jack Dongarra (co-chair, Oak Ridge National Laboratory)
Jeffrey Hittinger (co-chair, Lawrence Livermore National Laboratory)
John Bell (Lawrence Berkeley National Laboratory)
Luis Chacón (Los Alamos National Laboratory)
Robert Falgout (Lawrence Livermore National Laboratory)
Michael Heroux (Sandia National Laboratories)
Paul Hovland (Argonne National Laboratory)
Esmond Ng (Lawrence Berkeley National Laboratory)
Clayton Webster (Oak Ridge National Laboratory)
Stefan Wild (Argonne National Laboratory)

DOE/ASCR Point of Contact
Karen Pao

Abstract

This report details the findings and recommendations of the DOE ASCR Exascale Mathematics Working Group that was chartered to identify mathematics and algorithms research opportunities that will enable scientific applications to harness the potential of exascale computing. The working group organized a workshop, held August 21-22, 2013 in Washington, D.C., to solicit input from over seventy members of the applied mathematics community. Research gaps, approaches, and directions across the breadth of applied mathematics were discussed, and this report synthesizes these perspectives into an integrated outlook on the applied mathematics research necessary to achieve scientific breakthroughs using exascale systems.

Contents

Executive Summary

High fidelity modeling and simulation of physical systems is a critical enabling technology area for the U.S. Department of Energy (DOE), required for addressing some of the most challenging problems in energy, the environment and national security. The importance of continued advances in this area cannot be overstated. As a result, DOE is aggressively pursuing an exascale computing program that includes applied mathematics research focused on the unique challenges and opportunities present at this scale.

Exascale capabilities promise unprecedented potential for high fidelity, high confidence, and optimal solutions to complex multiscale, multiphysics problems at the heart of DOE's mission. Unlocking the potential of exascale computing requires the development of the next generation of computational models in order to satisfy the accuracy and fidelity needs for targeted problems. Specifically, more complex physical models must be developed to account for more aspects of the physical phenomena being modeled. Furthermore, for the physical models being used, increases in the resolution of the system variables are needed in order to improve simulation accuracy, which in turn places higher demands on computational hardware and software.

In order to meet DOE's science, design, and decision support needs, the computational capability of the fastest supercomputers must continue to grow. However, the transition from current sub-petascale and petascale computing to exascale computing will be at least as disruptive as the transition from vector to parallel computing in the 1990s. Driven mostly by power constraints, exascale-class machines (capable of 10^{18} floating-point operations per second or more) will see a massive increase in the number of computing units (into the millions), in the form of homogeneous cores or heterogeneous mixtures of multipurpose CPUs and specialized processing units. Memory and bandwidth will not increase as quickly as core count, and data transfer latencies will be further exposed. The shear number of components increases the potential for more frequent faults and failures.

In preparation for exascale systems, the DOE Office of Science Advance Scientific Computing Research (ASCR) program has sponsored a series of workshops leading to comprehensive reports on many of the challenges and opportunities. Nevertheless, the role of applied mathematics in the exascale computing effort has not been sufficiently explored. It is widely recognized that, historically, numerical algorithms and libraries have contributed as much to increases in computational simulation capability as have improvements in hardware. Mathematics permeates simulation; and the expected developments in computer systems will require a holistic reconsideration of all aspects of solving a problem, including formulation, discretization, scalable solvers, analysis, and software. In addition, applied mathematics has a role to play in exascale system software, providing algorithms and models for capabilities such as dynamic resource allocation.

This report details the findings and recommendations of the DOE ASCR Exascale Mathematics Working Group that was chartered to identify mathematics and algorithms research opportunities that will enable scientific applications to harness the potential of exascale computing. The working group organized a workshop, held August 21-22, 2013, in Washington, D.C., to solicit input from over seventy members of the applied mathematics community. Research gaps, approaches, and directions across the breadth of applied mathematics were discussed. This report synthesizes these perspectives into an integrated outlook on the applied mathematics research necessary to achieve scientific breakthroughs using exascale systems.

Concisely, the DOE Advanced Scientific Computing Research Program needs to take action to build a more explicit research program in applied mathematics for exascale computing. The necessary actions are summarized in five key recommendations:

1. DOE ASCR should proceed expeditiously and with high priority with an exascale mathematics initiative so that DOE continues to lead in using extreme-scale computing to meet important national needs.

2. A significant new investment in research and development of new models, discretizations, and algorithms implemented in new science application codes is required in order to fully leverage the significant advances in computational capability that will be available at the exascale. Many existing algorithms and implementations that have relied on steady clock speed improvements cannot exploit the performance trends of future systems.

3. Not all problems require exascale computation, and yet these problems will continue to require applied mathematics research. Thus, a balance is needed in the DOE Applied Mathematics Research portfolio that provides sufficient resources to realize the potential of exascale simulation while preserving a healthy base research program.

4. An intensive co-design effort is essential for success, where computer scientists, applied mathematicians, and application scientists work closely together to produce a computational science discovery environment able to exploit the computational resources that will be available at the exascale.

5. DOE ASCR must make investments to increase the pool of computational scientists and mathematicians trained in both applied mathematics and high-performance computing.

The majority of this report makes the case for these recommendations, based on a detailed explanation of the role of applied mathematics in scientific simulation and the associated research challenges motivated by the drive toward exascale computing.

Applied mathematics research is a critical component of the overall exascale computing enterprise in DOE. Enhancing the national capabilities in advanced mathematical modeling, numerical algorithms, and software will have a major impact on our future national research capacity and an international impact in the ever-increasing number of domains within which high-performance computing is, or is set to become, a core activity. It is essential that DOE make strategic investments now in high-performance computing algorithms and software in order to enable successful use of exascale resources in support of its mission and to safeguard our ability to continue to lead the world in this field.

Acknowledgments

The Exascale Mathematics Working Group express gratitude to many contributors to the process that led to the composition of this report.

We thank the domain scientists—Marv Adams, Bill Collins, Thomas Maier, Martin Greenwald, Jeremy Smith, and Paul Kent—who prepared presentations for our working group and patiently answered our many questions.

A large part of our information-gathering process was the workshop held in Washington, D.C. We thank Karen Pao and the staff at ORISE (Julie Webber and Keri Cagle, in particular) for their help with managing the logistics of the workshop. Of course, the content of the workshop came from the community, and we thank the many authors who submitted white papers, the workshop participants, as well as the scribes who helped record the discussions at the workshop: Jeff Banks, Jed Brown, Paul Constantine, David Gleich, Terry Haut, Matthew Norman, Rich Vuduc, and Qiqi Wang.

Despite the breadth of experience on the working group, we benefited from additional help in addressing some topics: Mark Shepard and Lori Diachin provided the section on scalable computational geometry and mesh generation, Ali Pinar and Janine Bennett contributed to the section on data analysis, and Bert de Jong and Chao Yang provided much of the text for the section on materials science. We are also grateful to Gail Pieper for her help in editing this report.

1 Introduction

The U.S. Department of Energy (DOE) is tasked with addressing some of the most challenging problems in energy, the environment, and national security. Addressing these challenges requires simulation of complex multiscale, multiphysics phenomena and may also involve mathematical optimization and uncertainty quantification to answer broader design and decision questions. However, even with today's mathematical algorithms and petaflop supercomputers, many extreme-scale science problems are still intractable.

A science-based case for investment in exascale computing has been established [21]. Over the past five years, the *Scientific Grand Challenge Workshop Series* has produced a string of reports on the open research questions in climate science [23], high energy physics [22], nuclear physics [30] and energy [29], fusion energy [27], materials science and chemistry [24], biology [26], and national security [28]. Advancing science in these key areas requires the development of the next-generation computational models to satisfy the accuracy and fidelity needs of targeted problems. The potential impact of these models on computational science is twofold. First, scientists will be able to account for more aspects of the physical phenomena being modeled. Second, increases in the resolution of the system variables, such as the number of spatial zones, time steps, or particles, will improve simulation accuracy. Both of these impacts will place higher demands on computational hardware and software.

To meet these science needs, the computational capability of the fastest supercomputers must continue to grow. However, the transition from current sub-petascale and petascale computing to exascale computing will be at least as disruptive as the transition from vector to parallel computing in the 1990s. Driven mostly by power constraints, exascale-class machines will see a massive increase in the number of computing units, whether homogeneous cores or heterogeneous mixtures of multipurpose CPUs and specialized processing units. Memory and bandwidth will not increase as quickly as core count, and data transfer latencies will be exposed further. The shear number of components—for instance, millions of cores—increases the potential for more frequent faults and failures. The proposed exascale architectures will present significant challenges for scalable software development and deployment.

Accordingly, the DOE Office of Science Advance Scientific Computing Research Program (ASCR) has started to prepare for the exascale computing challenges. Workshops have been held and reports have been written on many of the computer science challenges, including architectures [25, 32], operating and runtime systems [37], programming challenges [34], fault management [36], development and performance measurement tools [35], data management and analysis [33], and performance modeling and simulation [38]. The 2010 Advanced Scientific Computing Advisory Committee (ASCAC) Exascale Report [21] found that an integrated "co-design" effort will be essential for success, where system architects, application software designers, applied mathematicians, and computer scientists work closely together to produce a computational science discovery environment that fully leverages the significant advances in computational capability that will be available at the exascale.

Nevertheless, the role of applied mathematics in the exascale computing effort has not been sufficiently explored in and of itself. It is widely recognized that, historically, numerical algorithms and libraries have contributed as much to increases in computational simulation capability as have improvements in hardware. The expected developments in computer systems will place an even greater focus on algorithms as a means of increasing our computational capability. Significant new model development, algorithm redesign, and science application code reimplementation will be required in order to exploit effectively the power of exascale architectures. Some of these issues have been identified in previous reports [3, 31, 33], but, to date, no assessment has focused solely on the challenges and opportunities for research in applied mathematics for exascale simulation. This report addresses this deficiency by examining the role of applied mathematics research throughout the modeling and simulation process and by identifying important topics in need of more research.

1.1 Charter and Goals

In January 2013, ASCR formed an Exascale Mathematics Working Group (EMWG) to identify mathematics and algorithms research opportunities that will enable scientific applications to harness the potential of exascale computing.

The EMWG charter, written by the working group and approved by ASCR, was to do the following:

- Analyze potential gaps in current thinking about applied mathematics for the exascale;
- Identify new algorithmic approaches that address exascale challenges;
- Identify mathematics to address new scientific questions accessible at exascale, especially through integration across applied mathematics subdisciplines;
- Identify a holistic, co-design approach for applied mathematics exascale research that more directly involves a dialogue with application scientists and computer scientists; and
- Submit a report of the findings to the DOE Office of Science.

This charter reflected the desire of the working group to consider the breadth of applied mathematics activities necessary for extreme-scale science, from mathematical modeling through discretization and solvers to analysis and decisions.

1.2 Membership

The EMWG comprised ten research scientists from the DOE national laboratories:

Name	**Affiliation**
John Bell	Lawrence Berkeley National Laboratory
Luis Chacón	Los Alamos National Laboratory
Jack Dongarra*	Oak Ridge National Laboratory
Rob Falgout	Lawrence Livermore National Laboratory
Michael Heroux	Sandia National Laboratories
Jeff Hittinger*	Lawrence Livermore National Laboratory
Paul Hovland	Argonne National Laboratory
Esmond Ng	Lawrence Berkeley National Laboratory
Clayton Webster	Oak Ridge National Laboratory
Stefan Wild	Argonne National Laboratory

**co-chairs*

Karen Pao, an ASCR program manager for the applied mathematics subprogram, also participated as the ASCR point of contact for the working group.

1.3 History and Timeline

The EMWG was formed in early January 2013 at the request of William Harrod, director of the ASCR Division of Computational Science Research and Partnerships. The initial meeting of the working group occurred in late January via teleconference; the initial effort was to define a charter and to plan a course for gathering information. Most meetings were held as teleconferences, but the first face-to-face meeting of the working group occurred at the SIAM Conference on Computational Science and Engineering in late February 2013. To obtain information, the working group decided to solicit white papers from the applied mathematics community and to host a workshop to engage the community further. In addition, the EMWG decided to engage in a series of fact-finding teleconferences with domain sciences from Office of Science areas; these teleconferences occurred from April through June. Position papers were selected in May 2013; and the workshop, organized around these papers, was held August 21–22, 2013, in Washington, D.C. This report was written during the fall of 2013 and submitted to ASCR in February 2014.

1.4 Fact-Finding Meetings

Solvers and solver libraries are a mainstay of scientific computing and justifiably a core emphasis of applied mathematics research. However, mathematics plays a pervasive role extending "upsteam" in the modeling process. The mathematical formulation of the problem and its discretization are also important steps in simulation that impose constraints and challenges on the "downstream" linear and nonlinear solvers. Thus, the EMWG decided to investigate model formulation within the context of the problems facing DOE science application areas. The goal was to better understand the science needs—the open questions different science domains are trying to answer through simulation—driving the push to exascale, without limiting consideration to current practices. Many previous grand challenge reports focus heavily on building from the current state of the art without questioning whether that state is an artifact of the evolution of the field. The push to exascale not only may be an opportunity to change this but also may benefit from a fundamental rethinking of how the problems are posed.

The EMWG hosted six teleconference presentations by scientists representing the following areas:

Topic	**Speaker**	**Affiliation**
Nuclear (Fission) Energy	Marvin Adams	Texas A&M University
Atmospheric Science	William Collins	LBNL
Correlated Electron Systems	Thomas Maier	ORNL
Fusion Energy	Martin Greenwald	MIT
Biofuels	Jeremy Smith	ORNL
Materials Science	Paul Kent	ORNL

Perspectives formed from these discussions are the basis of Section 2 of this report.

1.5 Workshop

To stimulate a dialogue with the greater applied mathematics community, in May 2013, the EMWG issued a call for position papers on exascale computing research challenges in applied mathematics. Seventy-five position papers were received, and from these forty were selected for presentation. Workshop details, including the position papers, agenda, and attendees, are provided in the appendices. Electronic versions of the position papers are available for download from https://collab.mcs.anl.gov/display/examath.

The EMWG's Exascale Mathematics Workshop was held August 21–22, 2013, in Washington, D.C., with over seventy DOE laboratory researchers, academics, and government program managers participating. Several members of the European applied mathematics community were also present.

Position papers addressed topics such as scalable mesh and geometry generation, multiphysics and multiscale algorithms, in situ data analysis, adaptive precision, asynchronous algorithms, optimization, uncertainty quantification, and resilience. Each position paper was allotted ten minutes for presentation, and a substantial amount of time was devoted to group discussions about the ideas and issues raised by the presenters. After the workshop, a web-based survey was created to obtain additional feedback from the workshop attendees.

1.6 Report Organization

In the following section, we consider three motivating examples of the types of extreme-scale science problems that exascale computing may enable researchers to address; these exemplar science needs represent the forces from above that will affect the mathematics involved in scientific simulation codes. In Section 3, we briefly review the challenges of exascale computing as imposed by the expected changes in computational hardware; these challenges represent the forces from below on the mathematics involved in scientific simulation codes. We use a top-down analysis in Section 4 to identify research opportunities in applied and computational mathematics for exascale computing. The interdependencies between mathematics research and other exascale computing research activities are discussed in Section 5. The report concludes in Section 6 with findings and recommendations. Information on the workshop that informed this report, including the submitted white papers, workshop attendees, and workshop agenda, are provided in the appendices.

2 Motivating Science and the Role of Applied Mathematics

The science challenges that motivate the need for exascale-class computing resources have been well-documented [22–24, 26–30]. Two common themes of these science challenges are the extreme ranges of temporal and spatial scales and the complex nonlinear interactions of multiple physical processes. Predictive simulation capabilities are the goal, so that computational results can be used not only to increase scientific knowledge and understanding but also for design and decision. For context, we briefly review three such science areas of relevance to the DOE—combustion, climate, and materials—and highlight the associated mathematical challenges within each area.

2.1 Combustion

One area where exascale computing can make significant impact is the design of next-generation combustion systems such as high-efficiency, low-emission diesel engines that can burn new biodiesel fuels. On the surface, modeling the combustion process in a diesel engine involves the simulation of high-pressure turbulent reacting flow in a complex moving geometry. While this is certainly a requirement, a number of additional physical processes need to be modeled in order to simulate a diesel engine. Fuel is injected into the engine in a high-pressure, high-velocity liquid jet. The dynamics of this jet, which plays a critical role in engine performance, is a complex multiphase phenomenon where extremely fine-scale effects play a key role in the breakup and atomization of the fuel. The combustion process forms particulate soot as an intermediary in the combustion process. The formation and subsequent burnout of soot are other multiphase effects in which molecular-level processes govern the behavior of the system. Because soot is optically thick, radiative processes enter into the picture as well, combining with conductive heat transfer to the walls to define the thermal environment within the cylinder. High-fidelity simulation of this type of system is beyond the capability of current petascale systems.

These elements represent one aspect of diesel engine simulation; however, a number of other issues also play a critical role in predictive simulation of an engine. Diesel fuels are complex hydrocarbons whose chemical, thermodynamic, and transport properties are needed to perform a simulation. Reaction kinetics and thermodynamic properties for these types of complex molecules are not well known. Furthermore, even if the kinetics were fully understood, simulations with a comprehensive chemical mechanism would be infeasible. Transport properties are also not well understood, particularly in the high-pressure environment associated with diesel combustion. Quantifying the fidelity of a diesel engine simulation will require detailed uncertainty quantification to elucidate the uncertainty in predictions resulting from uncertainties in the fluid properties used in the simulation. These issues are further complicated by the need to model a variety of candidate biodiesel fuels in addition to traditional diesel fuel. Ensembles of simulations will be required in order to link experimental data to fuel properties and improve the predictive capability of simulations.

The simulation of a diesel engine is a complex multiphysics problem that needs to incorporate the effect of uncertainty across a range of different submodels and establish linkages between models and experimental data. However, the ability to perform simulations of a diesel engine with quantified uncertainty is only one step toward the actual goal, which is the design of a better engine. Designers would like to find optimal designs for fuel injectors and their placement in the cylinder, the shape of the cylinder bowl, and the placement and geometry of valves. These design problems are multiobjective: they need to balance fuel efficiency with emissions across a range of potential fuels and operating conditions. In addition, there are inherently stochastic because of cycle-to-cycle variability in the engine and of variations in the fuel characteristics. Solving these optimization problems will require methodologies for constructing rich hierarchies of models of quantified fidelity, combined with optimization algorithms that can utilize models of varying fidelity during the optimization process. The issues in diesel engine design are not simply computational power. A rich set of new mathematical tools is needed to enable the design of next-generation engines.

2.2 Climate

Climate modeling is another application where exascale computing has the potential to make significant impact. At a basic level the goal of climate modeling is to estimate the response of global temperature to increases in greenhouse gases. The full complexity of the problem becomes manifest when one tries to quantify how the climate system would respond to an increase in temperature. Climate scientists would like to answer questions such as what temperature rise is required to trigger a major climatic event (e.g., melting of the Antarctic ice cap or an irreversible shift in ocean circulation); how extreme weather patterns will change; and how large stores of carbon will respond to global warming.

As with combustion, answering these questions is not just a matter of harnessing more computer cycles; substantive mathematical advances are needed to address these problems. Climate models are complex multiphysics problems. They combine models for atmospheres, oceans, ice sheets, land surfaces, and the biosphere. Each of these models poses a challenging mathematical problem in its own right. In many cases, asymptotic convergence of the models has not been established. Furthermore, important questions arise about how to couple these models computationally. What are the key requirements to ensure that the combined model produces a stable and accurate prediction? How do errors in one model impact the fidelity of other components of the model? How accurately must each component be treated to ensure the fidelity of climate predictions?

Of equal if not greater importance is the multiscale character of many of the models used for climate simulation. In many cases, the basic physical processes are understood at small scales, but reliable techniques for representing those processes are larger scales are not known. For example, the fundamental processes for moisture physics in clouds are well understood, but it is not known how to represent ensembles of clouds at the scales required for climate simulation. There is no analog of statistical mechanics for clouds. Currently, it has proven challenging to link models of processes operating at the scale of climate models to benchmark models and measurements of these same processes operating at their native scales. The lack of systematic methodology for deriving representations of key processes as a function of scale is a fundamental barrier to progress. Developing the mathematical tools to address these types of scale-dependent models will require significant advances. Many of the processes that need to be modeled are highly nonlinear, and there is often no clear scale separation. Consequently, models based on a Markovian assumption will not be adequate. Models that can represent the physics across a range of scales are likely to be stochastic and include memory effects.

Another mathematical challenge in climate modeling is how to most effectively utilize observational data to improve predictions. Can we develop data assimilation schemes that improve model performance? What does a given set of observations tell us about the underlying physical process? What are the most effective quantities to measure to understand the connections amongst the different components of an earth system model? Answering these questions will require the development of new ideas at the interface of Bayesian statistical analysis, sampling methodologies, and optimization.

Climate models are not only used for basic scientific studies; they are also used for assessments needed by policy makers. However, precise deterministic prediction of long-term climatic trends is not feasible. In this type of setting, a single computation is not sufficient. Rather, climate scientists need to quantify the potential range of possible behaviors. In some cases, the goal is not an estimation of mean behavior but an assessment of the possibility of rare but catastrophic events. Obviously, these types of studies need to include estimates of uncertainties in the predictions.

2.3 Materials Science

The development of new novel materials plays a key role in solving technological challenges in areas such as artificial light harvesting to produce liquid fuels, energy storage in next-generation battery technologies, metal organic frameworks, zeolites, and organic photovoltaics. Advances in computational materials approaches are making inroads predicting material properties, identifying novel and potentially useful materials, and guiding the functional materials design process at an atomistic scale using petascale-computing resources. The need for exascale within computational materials sciences is driven by the need to predict and understand the behavior of new materials from the atomistic scale to the device level itself. Computational materials science at exascale will be key to enable advances in high-tech materials that will move us toward a sustainable, safe, and renewable energy future.

Predicting the behavior of heterogeneous materials with significant structural disorder and chemical complexity in macroscopic devices requires the modeling of emergent (mesoscale) properties and processes that bridge the many length (nanometer to microns) and time (femtoseconds to minutes) scales. Modeling the emergent properties and the multiphysics nature of various processes of complex systems at disparate length and time scales calls for a multiscale approach that can describe transport of ions and electrons, synthetic self-assembly of structures, electrochemical reactions at interfaces, heat generation and transfer, and structural deformation and stress. In order to bridge various length scales, it is essential to link continuum and other microscale models with atomistic and even electronic descriptions by providing correct up-scaling of interactions for coarse graining as well as down-scaling to perturb nanoscale and electronic environments.

Furthermore, in order to predict the transient behavior of materials, such as their structure and electrical, chemical, and thermal properties, it is essential to understand their behavior under real-world conditions. For example, modeling the behavior of lithium from the nanoscale to the microscale during the charging and discharging cycles of lithium-ion batteries can provide insight into the modes of failure and degradation of battery materials and can drive the design of better batteries. charge carrier diffusion, crack formation, and significantly longer timescales. Although a number of mathematical models have been developed to accomplish these goals, the numerical solution of the underlying equations in these models remains challenging.

Accurate approaches for describing the large, complex, heterogeneous nature of materials and the physical processes at the mesoscale are expected to be a truly exascale computing challenge [24]. To capture mescoscale properties of complex materials requires scientists to study systems consisting of millions of atoms. Another driving force in computational materials science requiring exascale resources is the enormous size of the search space from which optimal materials can be chosen. If one considers hundreds of thousands of potential materials that each need to be modeled accurately using teraflop or sub-petaflop simulations, the need for exascale becomes clear. The many-body nature of microscopic models makes the complexity of the computation grow rapidly with respect to the number of degrees of freedom. For ground state calculations, approaches based on density functional theory typically scale cubically with respect to the number of atoms, n. The scaling for wavefunction methods, such as the coupled cluster method, is even higher. For excited states calculations, methods for both extended systems and molecules have at least $O(n^4)$ complexity and in many cases can go up to $O(n^6)$. For $n = 1,000$, which is still relatively small, this complexity amounts to $O(10^{18})$ operations for a single calculation.

As an example of the many mathematical issues that enter into these multiscale, multiphysics materials problems, consider coupling a microscopic model to an atomistic or continuum model using a microscopic simulation to fit or estimate parameters contained in a higher-level model. The estimation process may require solving a system of tightly coupled nonlinear differential/integral equations iteratively. It may also require collecting information from multiple instances of microscopic simulations that can be carried out in parallel. Furthermore, uncertainty quantification and sensitivity analysis are important tools that could be brought to bear for tuning model parameters and making them adaptive to configuration and environment changes. Iterative solver acceleration techniques that can take advantage of physics-motivated preconditioner are highly desirable for solving both the coupling equations and nonlinear equations used in a microscopic model. In order to elucidate the dynamic behavior of the material, efficient and stable time-evolution schemes are necessary. In order to bridge the gap among different scales, multiresolution and multiscale methods based on asymptotic expansion, coarse graining, and statistical sampling are frequently used. All these mathematical techniques must be able to take advantage of the vast amount of computational resources and extreme concurrency available at the exascale.

3 Challenges at Exascale

Exascale will provide the computational power needed to address the important science challenges in DOE's mission, but that capability will come at an expense of a dramatic change in architectures. Numerous reports over the past five years have documented the technical challenges and the non-viability of the existing computer designs to reach exascale [25, 32, 57]. For context, we briefly summarize these challenges here.

Power: Power is the driving force behind the changes in supercomputer architecture. In some sense, exascale computing should really be thought of more as "low-power, high-performance computing." To continue to design supercomputers using standard commodity technologies is not sustainable; the power requirements of such a machine rapidly become prohibitive [57]. The goal has therefore been set to achieve exaflop performance with a power limit of 20 MW. This restriction has direct implications for the structure and organization of the hardware components as well as algorithms. It is conceivable that the energy used by a simulation may replace the CPU time as the cost metric for supercomputer use; hence, numerical algorithms may need to become more power-aware.

Extreme Concurrency: From hand-held devices to supercomputers, processor clock speeds have stagnated because of power density limitations. Instead, increased performance is being obtained by increasing the number of processing elements on a chip (multiple cores) and supporting threading. It is estimated that exascale machines will have two to three orders of magnitude of parallelism over petascale computer levels, with much greater parallelism on nodes than is available today. The bulk-synchronous execution models that dominate today's parallel applications will not scale to this level of parallelism. New algorithms need to be developed that identify and leverage more concurrency and that reduce synchronization and communication. One approach will be through dynamically scheduled task parallelism; but this will introduce a new challenge, reproducibility, that will make determinations of code correctness more difficult.

Limited Memory: Without a significant change in technology, memory density is not expected to increase at the same rate as the number of processing units. Again, power is a limiting factor; current volatile RAM technology, for example, consumes a great deal of power to maintain its state. Thus, while the amount of memory per node will increase, the amount of memory per core will decrease. Many current algorithms will thus be memory constrained and will need to be redesigned to minimize memory usage.

Data Locality: Similarly, memory bandwidth is not expected to increase at the same rate as the number of processing units. Consequently, on-node memory bandwidth will increase, but the bandwidth per core will actually decrease. Interconnect transfer rates are also not expected to increase at the same rate as the number of cores. In addition, the energy used for a double-precision flop is expected to decrease by roughly an order of magnitude, which will expose differences in the energy cost not just of off-chip data motion but of on-chip transfers as well. Future systems may use a variety of different memory technologies including nonvolatile memory, stacked memory, scratchpad memory, processor-in-memory, and deep cache hierarchies to try to ameliorate some of these challenges. Algorithms will need to be more aware of data locality and seek to minimize data motion, since this will be a more significant energy cost than will computation.

Resilience: Because of the shear number of components, hardware failures are expected to increase on exascale computers. Traditional checkpoint-restart recovery mechanisms are too expensive in terms of both the time and energy with bulk synchronization and I/O with the file system. Such global recoveries could conceivably take more time than the mean time between failures. Local recovery mechanisms are required that leverage the mathematical properties of the algorithms in the application. In addition, efforts to reduce power by computing with lower threshold voltages and other environmental disturbances may lead to more soft errors that may not be caught by the hardware. Increased fault rates will affect all hardware in the stack, but in particular applications may need to be fault-aware and use algorithms to make them tolerant to certain types of faults. The nondeterministic nature of failure and recovery, if occurrences are sufficiently frequent, will lead to nonreproducibility and make code correctness difficult to assess.

These are the key architectural changes expected to be necessary to build an exascale machine. Such architectural changes will force changes throughout the software stack in ways that cannot be completely hidden from the application and its associated numerical algorithms. Through model and algorithm development and design, mathematicians will need to address the new constraints these changes will affect. Particular constraints include the presence of distinct computing/architectural layers, leading to multiple levels of parallelism; severe penalization of data motion across architectural layers; the lack of hardware resiliency (in the form of both soft and hard errors) in some or all of the computing layers; the maximum exploitation of asynchrony in the implementation; the utilization of mixed-precision floating-point operations; and the maximization of the operational intensity.

4 Current and Future Research Directions

Several scientific applications within the DOE mission space require resources at the exascale (and potentially beyond). As demonstrated by the examples in Section 2, some of these needs arise from the desire to solve on larger-scale simulation domains, to solve for longer simulation times, or to solve with greater accuracy or resolution of finer spatial scales. Other needs stem from the desire to add additional or more detailed physical phenomena to increase the physical fidelity. Still other requirements stem from the need for meta-analyses such as sensitivity analysis, uncertainty quantification, and mathematical optimization.

What must be recognized, however, is that mathematics permeates the activities from the formulation of the problem to the analysis of the results (and, in fact, beyond). The realm of mathematical research necessary to make exascale computing a successful endeavor is not merely isolated to numerical solvers as implemented in software libraries. Computers do not "solve physics"; after all, computers fundamentally perform only a small set of logical operations.

Physical principles and problems are first expressed as mathematical models that are not, in general, in a suitably discrete algebraic form. Thus, these models must be discretized, typically leading to coupled nonlinear algebraic systems of equations, which then require robust numerical solvers. Efficient analysis of the resulting discrete solutions and verifying their correctness both require the application of additional mathematical techniques. Thus, analogous to the concept of the *software stack*, there is effectively a *mathematics stack* for simulation:

- Problem Formulation, or *Defining the question(s) to be answered*
- Mathematical Modeling, or *Expressing the problem mathematically*
- Discretization, or *Expressing the mathematical model discretely*
- Scalable Solvers, or *Solving the discrete system*
- Data Analysis, or *Understanding the results*
- Resilience and Correctness, or *Trusting the results*

In addition, there are common operations required for system management, such as dynamic resource allocation, that can be posed mathematically, for example, as optimization problems.

We will use this framework to organize our discussion of potential research directions for exascale computing. We emphasize that models and associated algorithms are not selected in isolation but must be evaluated in the context of the intended computer hardware environment. Specifically, we will discuss each of the above topics, the ways in which the challenges introduced by exascale architectures hardware (Section 3) will need to modify the current approaches to each, and some promising ideas that can address some or all of the exascale challenges.

4.1 Problem Formulation

Exascale computers offer a dramatic potential to change the questions we ask, improving our simulation capabilities from providing a single solution for a given set of boundary and initial conditions to providing an optimal solution with error bars. Future leadership-class computers will offer several orders of magnitude in potential performance improvement. How to best use this increased capability varies greatly across problem domains of interest to DOE. In some areas, the entire exascale system can be consumed by increased fidelity of a single forward simulation, whether that is through increased resolution (e.g., DNS of turbulent flows) or through more physically accurate (and complex) models that perhaps were previously considered infeasible. In other areas, forward simulations are already efficient and high-fidelity, leading naturally to the next simulation maturity levels [63] of optimization and uncertainty quantification (UQ), as depicted in Figure 1. Because of the new challenges and opportunities provided by these latter use cases, we choose to discuss them in more detail; higher-fidelity forward simulations will still represent a significant component of the workload on exascale machines, requiring very fast turnaround and support for new formulations, but this use case is better understood.

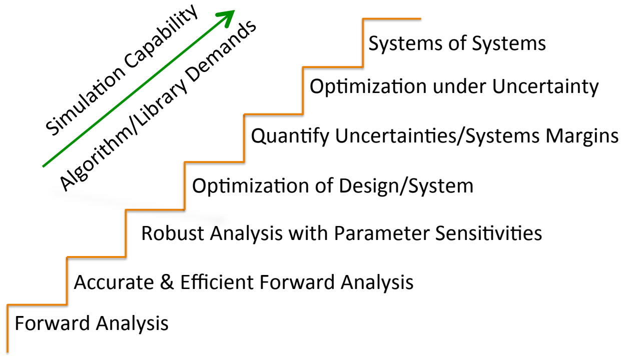

Simulation Capability
Algorithm/Library Demands
Systems of Systems
Optimization under Uncertainty
Quantify Uncertainties/Systems Margins
Optimization of Design/System
Robust Analysis with Parameter Sensitivities
Accurate & Efficient Forward Analysis
Forward Analysis

Figure 1: One depiction of the relationship between simulation capabilities. Each stage requires greater performance and error control of prior stages.

Mathematical optimization and UQ will increasingly be used in the exascale era, especially formulations in which these broader problems are tightly coupled to the underlying forward simulation model. Instead of UQ and optimization implemented as *outer loops* around the traditional forward simulation, techniques more tightly coupled to the forward solution strategy could provide opportunities for reuse, communication hiding, and even vectorization across multiple solutions. However, such formulations demand more from the underlying forward problem solvers, for example, leading to problems with multiple, simultaneous right-hand sides or to families of related linear systems with similar structure and spectral properties. In order to impact future codes, research is required now to develop, explore, and understand the myriad algorithmic design trade-offs.

4.2 Mathematical Modeling

With the goals of simulation well-defined, the first challenge is the mathematical formulation of the problem. In the context of scientific simulation, this necessitates the formulation of one or more mathematical models of the physical processes that dictate the physical system behavior. These physical laws and phenomena are expressed as well-posed systems of equations. In many simple cases, these equations are well-established (e.g., Navier-Stokes, Maxwell's equations); in more complex problems, a suitable mathematical model may be an open research question. As highlighted in Section 4.1, exascale will also bring increased scope for optimization and uncertainty quantification and mathematical formulations of the questions asked in optimization and uncertainty quantification. Selecting the appropriate mathematical model of the physics for and the level of coupling with these higher-level algorithmic demands is itself a modeling challenge.

4.2.1 Modeling Physical Processes

Within the DOE mission space, mathematical models often involve coupled physical phenomena—they are inherently *multiphysics*. In addition, there are many potential levels of description from the atomic up to cosmological scales. The level of fine-grained physical fidelity in models depends on the importance of the details of the finest-scale processes on the macroscopic time and length scales, but it is often limited by the available computational resources.

In the DOE applications that drive the need for exascale computing, nonequilibrium effects at the atomic scale and microscale (e.g., non-Maxwellian distributions of particles in plasmas, or cracks and voids in materials) are important. Fundamentally, we have classical and quantum models of atoms and molecules and could, in principle, attempt to simulate from this scale. However, such a model is an N-body problem with far too many degrees of freedom to simulate at macroscopic (engineering) scales, even with exascale resources. To manage the level of complexity and scale disparity of first-principles models, one must resort to dividing and conquering the scales by formulating reduced (or coarse-grained) models that target the appropriate level of description for a given set of dynamics. This approach naturally leads to *multiscale* or *scale-bridging* models, in which a coupled hierarchy of models is considered. Coarser-grained representations can be coupled sequentially or concurrently with finer-grained ones. Many approaches are available to derive such coarse-grained descriptions; we provide examples later in this section.

The advent of exascale computing is an opportunity to rethink the formulation and implementation of mathematical models to simulate physical systems. Exascale resources will make realizable the use of some models that were previously considered intractable (i.e., more complex, but more physically correct models). Systematic techniques that can construct coarse-grained models when such models are unknown can lead to stochastic partial differential equations, a field in which many opportunities exist for numerical algorithms research. Multiscale models that incorporate and couple descriptions across scales will also become more prevalent. Such multiscale models may provide novel opportunities for accelerating numerical solution by leveraging the many levels of description. We will discuss in more detail these topics, as well as the trade-offs between particle and continuum representations. Before proceeding, however, we comment on the limitations on mathematical models imposed by physical constraints.

Models must respect the physics. The mathematical properties of the models that make them difficult to solve numerically and in parallel most often derive from the physics. Hence, when considering a suite of model formulations for a given physical problem in the context of exascale computing, one must be careful not to trade physical relevance for parallel expediency. For instance, the propagation of information through a system is dictated by the underlying physics. If the details of this propagation are important, they must be resolved. If the details are unimportant, other mathematical models or numerical techniques can be used, but these models still capture the correct physical asymptotics.

A classic example is conductive heat transfer, which is typically described macroscopically by the parabolic heat equation. In a parabolic model, information propagates with apparent infinite speed. Numerically, the discrete system is globally coupled. Of course, this is an asymptotic approximation. Physically, the information propagates at a finite speed, but it appears effectively infinite over the scales considered. Alternative hyperbolic-relaxation models exist; but in order to take advantage of their locality (which benefits their parallel implementation), either the very fast time scales would need to be resolved (an expensive proposition) or a clever asymptotic-preserving scheme would be required to step over these fast time scales (generally a nontrivial exercise).

Alternative models may have advantages over those commonly used today, but the trade-offs need to be considered carefully. In the end, important physical processes must be respected, a requirement that constrains the behavior of the solutions to the mathematical models and ultimately the numerical techniques used to approximate these solutions.

Scale-bridging models. A class of mathematical models particularly suitable for exascale computing is scale-bridging algorithms. Such algorithms attempt to bridge disparate time and length scales in various ways, while at the same time avoiding a brute-force approach that would render the problem unmanageable. By nature, these scale-bridging algorithms exploit the separation of scales to devise optimal formulations at different levels of description of the problem. This naturally leads to a layered or hierarchical problem description, which can be beneficial when matched with the expected hierarchical nature of upcoming exascale computing architectures.

Hierarchical algorithms exploit nested levels of description (or layers) for solving multiscale problems. These layers may correspond to different description levels of the same physical system or to descriptions of different (but coupled) physical systems. Furthermore, the hierarchy of models may be applied globally across the simulation domain or locally, as in adaptive mesh and algorithm refinement, to restrict consideration of the finest scale to only those regions where such a description is important. The benefits of a layered algorithmic arrangement for exascale computing originate in the expected layered architectural arrangement of upcoming exascale computers. Often, different layers of a hierarchical algorithm will require vastly different computational resources. This requirement, in turn, allows one to target those levels of architectural parallelism that are most suitable for the description of interest.

From a solver standpoint, exascale computing will demand as much asynchrony as available. This, in turn, will demand both fine partitioning of the algorithm into simple tasks or kernels and taking full advantage of modern task-scheduling approaches such as directed acyclic graphs, which can automatically and on the fly schedule tasks according to prespecified dependencies among different tasks. In the context of hierarchical scale-bridging algorithms, however, partitioning and asynchrony as key organizational principles for the implementation of any given algorithm are not necessarily in conflict with a tight-coupling solution strategy. In particular, the layered arrangement of hierarchical algorithms, together with careful orchestration of the nonlinear solution strategy via nonlinear enslavement, allows the consideration of each layer as a separate entity from an implementation standpoint.

In general, the choice for the less computationally intensive layers in the algorithm will be fairly unconstrained by exascale architectures, since these will not be dominating the overall performance of the algorithm. However, the constraints imposed by exascale architectures will strongly influence the choice of representation for fine-scale physical models, which will represent the bulk of the computational work. As before, the general principles to consider are the ability to achieve fine-grained parallelism, resiliency (to both soft and hard faults), asynchrony, and floating-point precision. Based on these considerations, a general design principle for the representation of fine-scale physical models is minimum coupling between degrees of freedom.

Coarse graining. Hierarchical scale-bridging models require a method for coarse graining to arrive at a systematic hierarchy of models. In some cases, we know how to derive such a hierarchy; in other cases, the scale-bridging is more ad hoc and potentially prone to errors and inconsistencies between the levels of the description. More research on systematic techniques is required for well-posed hierarchical models. However, the need for coarse graining is more fundamental and needed for a wider range of problems.

Coarse graining is ubiquitous in numerical modeling and will continue to play an important role at the exascale. Whenever it is impossible to represent all of the degrees of freedom in a problem, one must resort to a coarse-graining approach. Classical partial differential equations describing fluid flow, solid mechanics, plasmas, and a variety of other problems are coarse-grained representations of particle systems. When these systems are solved numerically, one imposes another form of coarse graining, namely, the representation of continuous fields with finite-dimensional approximations. Analogous forms of coarse graining arise when modeling particle systems where it is infeasible to model all the particles needed to represent the system.

Another form of coarse graining arises when representing fine-scale behavior in terms of a coarse-grained representation. Examples of this type include mesoscopic models for fluids at micro-scales, models for crack propagation in solids, and models for clouds in climate models. The impact of unresolved degrees of freedom on the overall dynamics is the key question in these types of settings. In some cases, such as linear problems, a Galerkin-type approximation in which one simply truncates the effect of unresolved degrees of freedom is appropriate. However, this is not the case in general.

Many approaches to coarse graining have been developed, including averaging, homogenization, moment-based coarsening, renormalization group methods, and the Mori-Zwanzig formalism. While continued research is needed in general, we focus on two approaches that are particularly relevant to DOE: moment-based coarsening, which is intimately related to kinetic models, and the Mori-Zwanzig formalism, which is a more general strategy that often leads to macroscopic models expressed as stochastic partial differential equations.

Moment-based coarsening. In moment-based coarsening, coarse physics models are generated from a fine-scale model by recursive integration over one or several degrees of freedom. As a result, the coarse model features a reduced dimensionality but remains physically consistent at all scales. An example of such a process is the derivation of the hydrodynamic equations from the Boltzmann kinetic transport equation, where the integration is performed over velocity space.

The rigorous moment-based coarsening procedure provides well-defined restriction operators to communicate across the model hierarchy. Prolongation operators are not needed, since the fine description coexists with coarse ones. However, this coexistence raises questions regarding the impact of discrete consistency across the hierarchy, the preservation of conservation laws, and solver strategies. It also presents challenges in the context of mesh adaptivity: how to deal, for instance, with creation and destruction of patches in adaptive mesh refinement.

The development of hierarchical algorithms via moment-based model coarsening has many advantages for exascale computing. Coarse and fine models communicate only via moments, which live in a much-restricted dimensional space (e.g., 3D vs. 6D in most kinetic transport applications) and thus offer immediate benefits from the data motion standpoint. Asynchrony may be exploited in the way coarse and fine models are coupled; for instance, one can predict a fine closure and advance the coarse problem while computing a closure correction from the fine model. Moment-based models naturally allow the use of mixed precision, since coarse and fine models can use different floating-point precision; typically, the fine model is able to use a lower precision without overall loss of accuracy, since the effects of a lower precision are ameliorated by the moment integration procedure. Moreover, coarse and fine models may use completely different representations, targeting the most beneficial aspects of the intended architectural layer.

Stochastic systems. In a more general setting, the Mori-Zwanzig formalism from nonequilibrium statistical mechanisms [69] provides insight into the impact of unresolved degrees of freedom on the resolved dynamics. Within this framework, unresolved degrees of freedom enter the evolution of the resolved degrees of freedom as noise terms. The importance of these noise terms depends on the system and the scales being considered. In continuum models for fluids, for example, the hydrodynamic variables representing mass, momentum, and energy, which are obtained from the underlying particle description of the fluid, fluctuate. The amount of fluctuation depends on the scale being considered. At macroscopic scales, this effect is often negligible and can be ignored; but at mesoscopic scales, it can have a significant impact on the dynamics. For example, at small scales, mixing resulting from fluctuations dominates diffusive mixing.

Thus, at a fundamental level, coarse-grained systems are inherently stochastic. Traditionally, this aspect of the dynamics is either ignored or modeled with a deterministic term that represents the average of the stochastic behavior. Turbulence models attempt to model the effect of unresolved eddies with a deterministic diffusion term. Typical Arrhenius models for kinetics are phenomenological average models of finer-scale behavior. Since exascale enable simulations at higher fidelity, one must be cognizant of the effects of unresolved dynamics. Capturing the full range of behavior of the system may require inclusion of stochastic terms in the model. When a clear separation of scales exists, these stochastic terms can be Markovian; however, in a general case, the noise reflects a complex interplay with the history of the system.

These issues are particularly important for multiscale or hierarchical algorithms. In such settings, representation of the noise at coarser levels in the hierarchy is essential for accurate coupling between levels. Failure to capture this coupling can destroy the accuracy of the model at both coarse and fine scales.

We must allow for models of coarse-grained systems and hybrid multiscale algorithms to contain stochastic terms. These types of coarse-grained models can typically be written formally as stochastic partial differential equations. Even in fairly simple settings, however, these equations fall outside the scope of the mathematical theory of stochastic differential equations. Fundamental work is needed to establish approaches to analyze the basic properties of such systems.

Particle-based versus continuum representations. Two dominant mathematical representations of the model degrees of freedom are often used: particle models, where the dynamics of individual or collections of discrete entities in space are followed in a Lagrangian fashion, and continuum models, where functions on a continuous domain are numerically represented by a finite-dimensional representation associated with a mesh. There are also hybrid techniques (such as particle-in-cell methods), where some fields are discretized on a mesh and others by particles. The particle models are generally more fundamental and often can describe a broader range of (nonequilibrium) behavior but are noisy and can become prohibitively expensive at macroscopic scales. Continuum models are not noisy, and their convergence properties are much better understood; but they can become prohibitively expensive if used at microscopic scales (e.g., continuum representations of kinetic equations such as Boltzmann).

In some ways, particle-based models are well-suited for exascale computing. Particles can be processed independently until synchronization occurs at the next time-step level, which provides arbitrarily fine-grained parallelism and allows the exploitation of asynchrony. Because of their statistical nature, particle models can effectively use single precision, since the accuracy impact of lower precision is much smaller than the associated statistical noise. The statistical nature of particles also makes them resilient to both soft and hard faults. With regard to soft faults, given that particle orbits are independent from one another, one may check for corruption at the individual particle level and decide a course of action (e.g., rerun the orbit or remove the particle) independently from other particles in the ensemble. To recover from hard faults, one only needs to produce a *statistically equivalent* local reconstruction of the particle distribution, which in turn allows for significant data compression. Methods in this category include Monte Carlo, smoothed-particle hydrodynamics, and particle-in-cell techniques.

However, particle models have potential limitations as well. While they can provide high concurrency, particle methods tend to be low-order accurate and, in explicit algorithms, may fail to provide sufficient operational intensity (operations per bit transferred) to get beyond the bandwidth limitations that prohibit full floating-point utilization of node; this limitation is removed with modern implicit methods, which allow particle subcycling. In addition, particle methods are not well-suited for all problems. If the problem is near-equilibrium, for instance, continuum formulations represent an extreme compression of the data and thus are much more efficient at macroscopic scales; one would not use a particle model where moment equations are sufficient. Furthermore, the stochastic and interpolation noise may degrade accuracy in long-term simulations, and particle models make it much harder (but not impossible) to enforce local and global conservation laws. For these properties, continuum models are best suited. Of course, the latter increase coupling among degrees of freedom and thus are more constrained with regard to the exascale design principles previously outlined.

These general considerations suggest that in the context of hierarchical scale-bridging models, an a priori attractive choice is the use of continuum models for coarse-grained physical descriptions and of particle models for fine-grained ones. Some well-known methods (such as particle-in-cell) are formulated in this way by construction. Others, such as Monte Carlo for radiation and neutron transport, have been recently shown to gain significant accuracy and efficiency when coupled in this way. This recipe is particularly attractive for a large class of problems of interest to DOE that require the solution of kinetic transport equations. Kinetic transport models have been successfully used for years via particle formulations that efficiently represent the high-dimensional kinetic phase space. Their moment descriptions, however, typically benefit for a continuum treatment. Potential applications include radiation transport, neutron transport, plasmas (e.g., thermonuclear fusion), aerosol transport in climate, and combustion.

4.2.2 Uncertainty Quantification

Uncertainty quantification is a broad term for a variety of methodologies, including uncertainty characterization and propagation, parameter estimation and model calibration, and error estimation. The common goal of these activities is to address a fundamental question, namely how do the uncertainties ubiquitous in all modeling efforts affect our predictions and understanding of complex phenomena? Examples include both *epistemic* (lack of knowledge) and *aleatoric* (intrinsic variability) uncertainties, which encompass uncertainty coming from inaccurate physical measurements, bias in mathematical descriptions, as well as errors coming from numerical approximations of computational simulations. Because it is essential for dealing with realistic experimental data and assessing the reliability of predictions based on numerical simulations, advanced research in UQ ultimately aims to address these challenges.

The motivating science applications, described in Section 2, involve systems that describe physical and biological processes exhibiting highly nonlinear, or even worse, discontinuous or bifurcating phenomena at a diverse set of length and/or time scales. Hence, simulating the entire complex system at the level of resolution necessary to represent this behavior accurately is extremely difficult for many problems of national interest. Moreover, predictive simulation of these systems requires significantly more computational effort than do high-fidelity deterministic simulations, particularly in the case of climate models, where both the subgrid closure approximations and the input data (coefficients, forcing terms, boundary conditions, geometry, etc.) are affected by large amounts of uncertainty. Even for these high-dimensional stochastic problems, simulation code and calculation *verification*, model *calibration*, *validation* and bias correction, and a complete quantification of all uncertainties are indispensable tasks required to justify a predictive capability in a mathematically and scientifically rigorous manner. Figure 2 illustrates how these tasks are tightly connected. Unfortunately, current approaches for implementing these processes remain computationally taxing, making accurate predictive simulation of most complex stochastic systems exceptionally difficult. Confidence in simulation results is typically founded on a mix of intuition and an extensive sensitivity analysis. As such, it is critical to organize and design computational simulations and physical experiments that ensure that the right type of data, and enough of it, is available not only to quantify uncertainties, but also to understand and ultimately reduce their effect on quantities of interest (QoIs).

As the complexity of these systems increase, scientists and engineers are relied upon to provide expert analysis and to inform decision makers about the behavior of and the risk associated with predictive simulations. Many UQ approaches have been developed, including random sampling (see [44] and the references therein), stochastic polynomial methods, such as interpolatory collocation approaches [62, 68], and Galerkin projections [6, 46], that have been extensively utilized on large-scale applications. However, numerous changes in scientific computing at extreme scales are expected to challenge the current UQ paradigm, wherein the stochastic loop is typically wrapped around a *black-box* simulation. Expected decreases in single-core performance and memory per core, massive increases in the number of cores, and the emergence of novel accelerator-based architectures will require the development of new methodologies that integrate uncertainty analysis into computational simulations [3, 19, 28, 31]. In working toward this goal, several challenges arise when applying UQ methodologies to the DOE mission science areas.

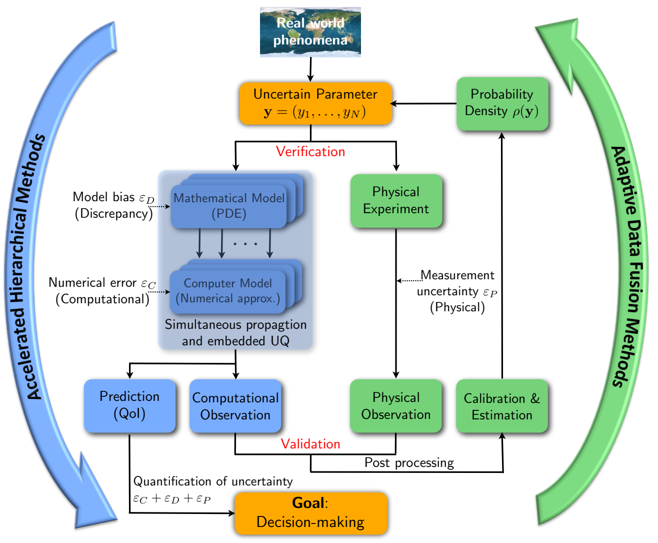

Real world phenomena
Uncertain Parameter $\mathbf{y} = (y_1, \dots, y_N)$
Probability Density $\rho(\mathbf{y})$
Verification
Model bias ε_D (Discrepancy)
Mathematical Model (PDE)
Numerical error ε_C (Computational)
Computer Model (Numerical approx.)
Simultaneous propagtion and embedded UQ
Physical Experiment
Measurement uncertainty ε_P (Physical)
Prediction (QoI)
Computational Observation
Physical Observation
Calibration & Estimation
Validation
Post processing
Quantification of uncertainty $\varepsilon_C + \varepsilon_D + \varepsilon_P$
Goal: Decision-making
Accelerated Hierarchical Methods
Adaptive Data Fusion Methods

Figure 2: Illustration of the unprecedented capability of exascale computing to assimilate predictions from computational simulations and data from physical experiments.

- Detection and quantification of high-dimensional stochastic QoIs with a specified certainty
- Reducing the computational burden required to perform rigorous UQ
- Efficient strategies for UQ that exploit greater levels of parallelism provided by emerging many-core architectures
- Systematic assimilation of the uncertainty in measured data for validating and correcting model bias, calibrating parameter interrelations, and improving confidence in predicted responses

To address these challenges requires a transition from currently used, nonintrusive algorithms and standard intrusive implementations to a truly architecture-aware, predictive capability. In what follows, we highlight several possible UQ research directions related to extreme-scale computing: adaptive hierarchical methods for high-dimensional approximation; multilevel methods for solution acceleration and complexity reduction; architecture-aware UQ paradigms; and adaptive and robust methods for combining computational simulations and experimental data.

Adaptive hierarchical methods for high-dimensional approximation. It is widely recognized that rigorous uncertainty quantification at the extreme scale will dramatically improve our understanding of physical and engineering problems [28, 31]. Indeed, a thousandfold increase in computing power would facilitate orders of magnitude more simulation realizations. However, enhancing the accuracy of stochastic QoIs requires the computational simulation to increase the number of *random variables* (dimensions), and expend more effort approximating the the solution in each individual dimension. The resulting explosion in computational effort is a symptom of the *curse of dimensionality.* To combat the computational cost, several methods that use sparse polynomial or sparse grid approximations in high-dimensional parameter, have gained considerable attention in the past decade (see [19, 48] and the references therein).

However, when the parameter space is truly high-dimensional or when the random solution exhibits steep gradients, sharp transitions or bifurcations, or jump discontinuities, all stochastic polynomial-based methods converge slowly or even fail to converge. Also, these approaches attempt to construct highly accurate, approximate solutions over the entire parameter domain (so that the approximation achieves the same accuracy everywhere); yet, building such a surrogate in the low-probability region of a joint probability density function (PDF) (i.e., using samples that contribute very little to the QoI) dramatically wastes computational resources. An ideal alternative is to construct solutions whose accuracy decreases as the PDF approaches zero, effectively approximating the solution only in the high-probability region. Moreover, the a posteriori PDF obtained when calibrating the input parameters can be highly irregular, not necessarily separable into a product of one-dimensional PDFs, making it extremely challenging to construct polynomial bases. To address these difficulties requires the invention of adaptive hierarchical approaches for low-discrepancy sampling with sufficient volumetric coverage of the input density; interpolation and approximation; and detection of events, in high-dimensional parameter spaces.

Advanced multilevel methods for solution acceleration and complexity reduction. In any UQ approach, the dominant cost of quantifying uncertainty in simulations lies in the solution of the underlying deterministic model. Indeed, many high-fidelity, multiphysics models can exhaust the resources on the largest machines with a single instantiation and thus are not practical for the most advanced UQ techniques. Moreover, future increases in computational resources will be accompanied by continuing demands from application scientists for increased resolution and the inclusion of additional physics. Consequently, new approaches are needed in order to decrease simulation computational costs within the UQ context. Such techniques will be required to harness both the underlying model hierarchy and the stochastic hierarchy.

Acceleration through multilevel methods using model hierarchies. Given the complexity of extreme-scale applications, multilevel surrogates will be needed that simultaneously utilize a hierarchical approach in both physical and parameter spaces. As opposed to existing work in principal orthogonal decomposition (POD)-type reduced-order models, more advanced hierarchical models will be needed involving variational methods for preserving important mathematical structures of highly resolved solutions; asymptotic expansions that take advantage of scale separation for highly accurate multiscale; and Mori-Zwanzig formulations, mean-field theory, and moment methods for high-order closure approximations. These methods can be used to develop multilevel formulations of the deterministic problem that will be used to elucidate common solution structures across multiple UQ levels, including structures induced by multiscale dynamics and scale separation. However, in order to minimize the total cost for the prescribed error, a general strategy must be developed to balance the contributions from approximation error in the stochastic space with model error in the deterministic space. Such a strategy will require analysis of both the deterministic model errors and the stochastic polynomial or sampling errors.

Acceleration through exploitation of the stochastic hierarchy. The bulk of the computational cost of extreme-scale computer simulations is usually associated with linear or nonlinear iterative solvers. One possible way in which the convergence of such methods can be dramatically improved is to "seed" the solver with a good initial starting point. Hierarchical UQ approaches produce multilevel sequence of approximations (e.g., interpolants), where, at each level, one introduces new sample points. Taking advantage of the hierarchical structure, one can accelerate the solution of the deterministic system at each level by using the stochastic approximation from the previous level to determine the new initial iterates. In essence, the iterative solver has to resolve only the correction at each level, resulting in a significant reduction of the computational burden. This idea will need to be extended in order to enable the use of existing preconditioners at each level to inform solvers at the next level of the hierarchy. New preconditioners also will be needed to accelerate the convergence of additional sampling/collocation points or polynomial bases. Incorporating these ideas into the multilevel framework, based on model hierarchies described above, will allow for further reduction in the total computational cost.

Architecture-aware UQ paradigms. Nearly all existing uncertainty propagation methodologies wrap around deterministic simulation codes. A typical example is nonintrusive implementations of traditional sampling-based methods that repeatedly call a deterministic simulation code for different values of the stochastic model inputs (usually according to a joint PDF). These approaches have been effective at producing software and algorithms relying on modest numbers of simulations that scale well on existing petascale architectures. However, future exascale computers are not likely to provide enough concurrency for *a thousandfold increase* in petascale sample evaluations for uncertainty propagation applied in this manner. Power and cooling limitations will favor compute nodes with dramatically increased floating-point capacity through increased node-level parallelism rather than increased clock speed or node counts [25]. Thus, increasing concurrent sample evaluation will require executing each sample on a smaller number of compute nodes or executing multiple samples simultaneously on each compute node.

In order to leverage the increase in node performance and other likely exascale characteristics, it may become beneficial to evaluate samples in parallel through a multilevel *embedded* propagation scheme whereby collections of samples are executed asynchronously and samples within each collection are propagated simultaneously at the node and processor core levels. The ability to embed portions of the "uncertainty loop" at the lowest levels of the simulation code requires replacing each scalar datum in a calculation with an array for the uncertainty representation of that datum, such as samples in a stochastic collocation-type method or polynomial coefficients in a stochastic Galerkin-type method. One possible path to accomplish this is to use code transformation techniques based on automatic differentiation [47]. Any operations on that datum can then be translated to parallel operations on the uncertainty array, both improving locality and exposing additional fine-grained parallelism. Since the messages for multiple realizations are incorporated into one message for the ensemble, total communication time should be reduced. Moreover, such an approach enables new algorithms that reuse data and calculations across uncertainty representations to reduce aggregate simulation cost, for example, reuse of mesh calculations that do not depend on uncertain input data and reuse of solvers and preconditioners across an ensemble. Of course, the concept of propagating multiple samples simultaneously at the scalar level of the simulation is predicated on the assumption that the code paths for these samples do not diverge greatly; otherwise no benefit or possibly a negative result is achieved. Careful research is needed that connects these ideas to high-level adaptive uncertainty propagation methods that decide when and how to group samples for co-propagation.

Adaptive and robust methods for fusing computations with experiments. Exascale computing will offer an unprecedented ability to assimilate the uncertainty in measured data for validating mathematical descriptions and correcting model bias, calibrating parameter interrelations, and improving confidence in predicted responses. Computer simulations can be used to quantify the uncertainties in complex physical systems, and physical experiments can be used to validate the computer simulations. Innovative adaptive and robust experimental design strategies are needed that intelligently gather data from both computer simulations and physical experiments in order to reduce variability in the estimate of the unknown parameters and the uncertainty in QoIs. However, these approaches must go beyond the standard Gaussian process modeling, or kriging [65], that require the solution of an inverse covariance matrix, which becomes computationally intractable as the parameter dimension increases. Possible research directions include the integration of generalized Bayesian techniques, as well as PDE-constrained parameter identification approaches, with the paradigms and approaches described previously. Such an integration would allow efficient generation of surrogate approximations for extreme-scale simulations for use within the calibration procedure, so as to reduce variability in the estimate of the unknown parameters and the uncertainty in QoIs. Of course, both approaches rely heavily on numerical optimization, either to determine the number of significant modes of a distribution or to assist in solving an adjoint problem. These techniques are discussed in detail in Section 4.2.3 below.

4.2.3 Mathematical Optimization

Mathematical optimization involves finding the best value(s) of an objective function, subject to constraint functions characterizing the feasible design/decision space. When physical or simulated phenomena are involved, these constraints necessarily include the space of realizable solutions from a simulation code. To date, optimization primarily has played the role of a sequential "outer loop," and hence research has focused on parallelizing the underlying linear algebraic operations within an optimization step and/or parallelizing the forward evaluation of objective and/or constraint functions. For many problems, the exascale will bring an abrupt end to savings based on such parallelism, especially as scalable evaluation of the objective and constraint functions becomes increasingly complex and difficult. Algorithmic-based approaches to fault tolerance are also expected to have an increased scope for optimization algorithms at the exascale.

Furthermore, the sustained increase in computational capabilities will enable the solution of new classes of optimization problems (see, e.g., Section 2) and bring mathematical optimization to new scientific and engineering domains. In particular, exascale resources open up new possibilities at the intersection of optimization and UQ in areas such as optimization under uncertainty, robust optimization, and optimization-based model calibration. Exascale computing will begin to provide the necessary resources to facilitate optimization and UQ for complex multiphysics codes; however, significant research is needed in order to design algorithms that address the challenges outlined in Section 3. Here, we consider four exascale research topics in mathematical optimization: concurrent-point methods; mixed-integer, simulation-based, and global optimization; multifidelity hierarchies; and robust optimization and optimization under uncertainty. Also considered is the optimal design and coupling of experiments.

Concurrent-point methods. A fundamental, unresolved challenge for optimization is the development and analysis of "concurrent-point methods." Such methods determine multiple, distinct (but possibly related) design/decision points for concurrent evaluation, where the evaluation may be done asynchronously or at differing levels of fidelity. Such methods represent a substantial shift from practices in traditional mathematical optimization, where optimization loops are inherently sequential, and objective and/or constraint functions (and the corresponding derivatives) are evaluated at a single point per iteration/loop step. Research to date on concurrent and asynchronous evaluation has focused largely on zero-order ("derivative-free") methods, which require more iterations than derivative-based counterparts; techniques are needed for more general, derivative-based methods. Analysis should provide approximation bounds for the complexity of these new algorithms, and classes of problems where concurrent-point methods can result in provable/substantial savings will need to be established. Related work includes *s*-step and communication-hiding Krylov methods, which can be viewed as solving a specific form of unconstrained optimization problem. Decomposition-based approaches may be a promising avenue (e.g., determining points in families of subspaces), provided that global reductions occur infrequently.

Mixed-integer, simulation-based, and global optimization. One example where such concurrent-point methods may admit excellent scalability is when there exists a combinatorial structure in the design/decision space. For example, when performing PDE-simulation-based optimization when discrete decisions are also present, relaxations corresponding to PDE-constrained, continuous optimization solves can be viewed as treelike structures. Disjoint leaves of such trees provide natural sources of distinct design/decision points for concurrent evaluation, but partitioning such combinatorial structures in order to obtain savings from the concurrent evaluations (e.g., by grouping related PDE solves and data structures in order to exploit reuse and minimize data movement) requires substantial algorithmic research.

Grouping related solves, whether optimization subproblems or forward simulations, may also fundamentally alter the tree structures encountered in global optimization. Finding a global solution—that is, the best of among all local minimizers—is an NP-hard task in general, but exascale may present new opportunities for global optimization algorithms with theoretical guarantees for special classes of problems.

Another potential way to reduce the number of iterations in an optimization loop is to exploit high-order derivatives of the simulation output with respect to the design/decision parameters. Current practices have focused on matrix-free application of Jacobians or Hessians; but for some problem structures, increased arithmetic intensity can be achieved by considering higher-order derivatives. However, research will be needed on scalable computation and/or application of these derivatives as well as concurrent adjoint calculations based on ensembles of related points. Efficiently exploiting sparsity structures also becomes increasingly critical as ensembles of derivatives and higher-order derivatives are considered.

Multifidelity hierarchies. As with UQ, research is also needed on optimization algorithms that can exploit multiple levels of fidelity and algorithmic-based approaches to resiliency. In the context of optimization, such methods acknowledge and embrace the principle that far from the solution, one need not perform expensive (e.g., because of high-fidelity or resilience considerations) evaluations. Optimization algorithms will thus need to select points for evaluation along with corresponding fidelity levels, possibly informed by knowledge (e.g., architecture-based, performance model-based) of the expense associated with performing that set of evaluations (due to data motion or otherwise). Work on multilevel optimization techniques has focused primarily on problems with an underlying grid structure, and determining appropriate coupling across a complex algorithmic hierarchy remains an open challenge. Potential adaptivity of the algorithmic hierarchies (e.g., as the set of active constraints changes or as a result of adaptive refinement) is also expected to present a challenge to scalable performance.

Robust optimization and optimization under uncertainty. Many problems of DOE interest require making decisions whose consequences cannot be fully determined. Uncertainty in data, parameters, and models can have deleterious effects on "optimal" solutions that do not account for such uncertainty, especially in nonlinear optimization problems where the constraint or objective values are highly sensitive to such uncertainty.

Fundamental differences in problems in this area can often be attributed to the way the uncertainty is modeled. In stochastic optimization the uncertainties are captured through distributions, whereas in robust optimization a min-max approach is followed using a compact uncertainty set. Determining the proper nesting order and level of coupling for the uncertainty and optimization (or min and max problem) hierarchies will likely be critical to achieving scalable performance.

Methods for stochastic optimization have natural tie-ins with ensemble-based UQ approaches and share similar challenges to scalability. For example, evaluating many scenarios concurrently can mitigate variance and uncertainty, but overall savings may result only if such fidelity is truly useful and if the scenarios admit scalable/resource-constrained evaluation. Provable guarantees, such as convergence and error bounds, for stochastic optimization methods using more general forms of ensembles (e.g., based on performance considerations) are needed. General strategies with looser dependence on the number of independent scenarios are also needed in order to enable scalable decision making and design under uncertainty.

Optimal design and coupling of experiments. As experiments become increasingly automated, operating computational experiments at leadership computing facilities and physical experiments at DOE user facilities in tandem is expected to become more frequent [21]. Such coupling can be characterized as computation-driven or physical-experiment-driven. Possible computation-driven scenarios include optimal design and model construction, and possible experiment-driven scenarios include experiment refinement based on computational analysis.

Currently, optimal design typically assumes a mathematical model for a physical system and employs numerical optimization on this model to identify an optimal configuration or set of parameter values; the resulting design is then validated by a physical experiment. Multifidelity methods use multiple mathematical models of varying fidelity and computational cost, alternating among the models to reduce the cost of finding an optimal set of parameters for the high-fidelity model; but again, validation through physical experiments is performed sequentially. With closer coupling between computation and physical experiment, we expect promising designs to be evaluated during the course of the optimization, with the experimental results informing subsequent optimization iterations. Realizing this capability will require mathematical innovation. It is possible that multifidelity methods can be adapted to this new context by treating the physical experiments as the highest-fidelity model, but they will need to be modified in order to account for experimental error and the typically more limited set of quantities that can be measured in a physical experiment versus a computational experiment.

In the construction of computational models using data from physical experiments, it is typically assumed that the physical data is collected and then the model is constructed. At best, a "design of experiments" methodology is used to determine which physical experiments to perform in order to provide the most useful data. With close coupling between computation and physical experiment, however, it becomes possible to take an *active learning* approach to model construction and choose which experiments to perform based on their expected impact on the accuracy of the model. Mathematical research in areas such as optimal sequential design of experiments and optimal stopping problems, at the intersection of optimization and uncertainty quantification, is required to facilitate this new paradigm.

4.2.4 Related Position Papers

Many position papers related to mathematical modeling were submitted to the Exascale Mathematics Workshop. These include papers on modeling physical processes [WP1, WP25, WP28, WP31, WP36], uncertainty quantification and data fusion [WP9, WP10, WP13, WP20, WP21, WP25, WP26, WP29, WP30, WP32, WP37, WP44, WP47, WP53, WP63, WP72, WP74, WPA1] and mathematical optimization [WP15, WP51, WP52, WP64, WP65, WP74]. Additional position papers incorporating these topics include [WP24, WP33, WP54, WP60, WP66, WP69, WP73] and position papers related to combinatorial and graph-based problems [WP42, WP43, WP48, WP49, WP67].

4.3 Model Discretization

Typically, the mathematical formulation of the problem cannot be directly solved by a digital computer, and so the problem must be approximated with a finite-dimensional representation. Continuous independent variables must be subdivided into discrete mesh points, cells, elements, time levels, etc. In the case of particle models, the spatial representation is already discrete, but the temporal independent variable requires discretization. In addition, the physical models, equations (operators), and dimensions may be split into submodels that are easier to discretize and solve. Especially in multiphysics applications, such splittings require careful coupling approaches to preserve temporal accuracy and to respect nonlinear processes. For multiscale models, coupling procedures are also paramount.

The choices made during the discretization will directly affect the properties of the resulting algebraic systems and will therefore affect the solvers used to obtain the approximate solution. Thus, discretization must be considered when addressing the challenges of exascale. Here, we discuss several of the exascale-relevant topics related to discretization, motivated both by the models requiring exascale resources and by the challenges posed by exascale architectures. Specifically, we address issues involved in coupling and partitioning, the advantages of high-order algorithms, adaptivity of mesh and models, and issues associated with computational geometry and mesh generation.

4.3.1 Model Coupling and Partitioning

The coupling of physically distinct physical models and/or of multiple scales will be a pervasive issue in exascale simulation. It is important to note that whether a *physical model* is weakly or strongly coupled depends on the time scale of interest. Thus, most physical models can be considered weakly coupled when fast time scales of the involved physics are respected. However, if integrating these systems over slower time scales is of interest, the same models may become strongly (and nonlinearly) coupled.

Accordingly, the solution strategy for multiphysics, multiscale models will depend largely on the physics of interest or the willingness to respect fundamental scales. Often, when well-verified models for distinct physical processes or scales have been developed for a particular purpose, computational scientists seek to reuse and combine these codes or modules in an attempt to model multiphysics or multiscale effects in a different regime. This type of composite discretization (which, following [56], we refer to as a *loosely coupled solution strategy*) is pervasive but can have stability or consistency problems, particularly in very long simulations. For strongly coupled systems, the particular loosely coupled solution strategy of choice may be either ill-posed or ill-behaved. Nonlinearly converging physics modules or models (a *tightly coupled solution strategy*) usually result in more stable and robust formulations. However, a global nonlinear solve will in general not be attractive at the exascale, and other approaches must be pursued. Furthermore, in scale-bridging applications, even tight coupling does not guarantee nonlinear stability. For instance, numerical truncation errors generated at different scales may amplify as these errors propagate through the hierarchy of models. For a recent survey of multiphysics modeling, associated implementation issues, and example applications, see [56].

Exascale computing provides an opportunity to rethink and to improve how models should be coupled across scales and physical processes. Various types of partitioning will be necessary for exascale simulation of multiphysics, multiscale models, and many opportunities arise for research into partitioning techniques. A critical issue is to develop tightly coupled solution strategies that ensure nonlinear convergence. Given the complexity of the anticipated exascale computers and questions about their reliability, verification of coupled models will require a much sounder theoretical understanding of the stability and accuracy of coupling techniques.

Partitioned algorithms. Partitioning can be geometric, operational, and model-based. In geometric partitioning, one splits the equations into separate subdomains (e.g., as in fluid-structure interaction [2]). In operational partitioning, one splits the equations or operators into subsets applied in some sequential or iterative fashion in (pseudo-)time (e.g., traditional operator splitting). In model-based partitioning, one segregates different code sets based on the physics model at hand.

Partitioning has the advantage that modular, efficient solvers for each domain or operator can be applied to that subproblem (e.g., an implicit treatment of a parabolic operator and an explicit treatment of a hyperbolic operator). In addition, partitioning can provide opportunities for task parallelism and reduced synchronization. Partitioning does not, however, necessarily imply a loosely coupled strategy. For instance, one can formulate tightly coupled nonlinear residuals by nonlinearly eliminating a (partitioned) physics module. If loose coupling is favored, the challenge again is to couple the subproblems in a way that accurately and stably captures the true nonlinear coupling of the original governing equations. Suitable compatibility conditions must be determined, whether in the form of modified boundary conditions or scale-bridging prolongation and restriction operations.

The preferred method of choice will generally be application dependent, where the strength and nature of the coupling must be considered. Some physical models may be more tightly coupled, and partitioning those models will require a greater degree of care than has been done previously in more ad hoc code coupling. Coupling may be *volumetric* or *interfacial*; in the former, the subproblems coexist in overlapping regions, while the latter couple through (possibly evolving) boundaries. Similarly, coupling may be *global*, where all models coexist everywhere in the domain (e.g., MHD), or *localized*, where models may be relevant on disjoint spatial (e.g., fluid-structure interactions) or temporal (e.g., nonlinearly switching models as the solution evolves) domains or on small regions of overlap (e.g., *hybrid models* that transition from fluid to kinetic models near material interfaces). Although the specifics may vary, in all cases the desire is to maintain consistency, accuracy, and stability while obtaining increased computational efficiency from the splitting. The challenge stems from the strength of coupling between the models/scales. Failure to respect strong coupling can lead to numerical difficulties. Proper partitioning is an important area for research as multimodel (multiphysics/multiscale) codes are designed for the exascale.

One topic, in particular, that deserves further investigation is high-order, partitioned time integration algorithms suitable for multiscale, multiphysics problems. Broadly, multimethod and multirate methods are two general approaches to dealing with multiple time scales. Implicit-explicit (IMEX) integrators, which include the many variants of multistage (e.g., Runge-Kutta), multistep (e.g., BDF), and hybrid (i.e., generalized linear methods [16, 17]) schemes, are perhaps the more familiar of the multimethod approaches. There are also multimethod and multirate spectral deferred correction (SDC) schemes [39], such as semi-implicit SDC [60] or multi-explicit SDC [12]. An alternative approach to dealing with stiffness introduced by fast time scales is based on the exponential propagation iterative methods [50, 51]. All of these methods partition the equations, operators, or spatial domain into subsets, each to be integrated by a time integrator best-suited for that subset; as such, each plays an important role in multiscale, multiphysics problems. Classically, for multimethod schemes, an implicit integrator would be used for a numerically stiff subset and an explicit integrator for the nonstiff part, thus allowing the entire system to be integrated at the larger, nonstiff time step. Of course, implicit discretizations trade stability for temporal accuracy, a useful trade-off when the details of evolution of the fast time scales are unimportant. Partitioned schemes provide systematic means to combine and interleave the results of the subintegrators to preserve accuracy and stability, including not just implicit-explicit coupling but also explicit subcycling (in the case of multirate schemes). Partitioned time integrators are and will continue to be important for solving multiphysics and multiscale problems. By partitioning the original model, these techniques may provide opportunities for task parallelism and asynchrony; but methods will need to be designed with minimal memory requirements, data reuse, and resilience in mind. Accurate long-time integration of multiscale, multiphysics models is another area requiring research.

Trade-offs will arise in the treatment of multiphysics and multiscale coupling; one solution will not be suitable for all problems. An important paradigm that should be followed, however, is that all problems should be considered "coupled until proven uncoupled." In other words, instead of starting with submodels and trying to patch them together, one should begin by considering the entire mathematical model and first understand the nature of the coupling within it. Any partitioned discretization must be consistent with this full model, and understanding the coupling in the model will help determine whether loosely coupled or tightly coupled solution strategies are required for different partitionings of the problem. The generation and propagation of numerical error caused and potentially amplified by operationally partitioning a strongly coupled nonlinear system must then be understood before using such an approach in an application. For scale-bridging applications, for example, a tight-coupling strategy is probably essential unless there is good understanding of the accuracy impact of a more loosely coupled approach. This is not, however, an unqualified endorsement of tightly coupled approaches. Much more research is needed into understanding how and when to partition nonlinear, multimodel systems accurately and stably.

Nonlinearly converged strategies. Given the multiscale nature of the problems of interest at the exascale, it will often be of interest to step over fast time scales to evolve the system on a slower manifold. This approach will likely lead to strongly coupled nonlinear systems, which will demand nonlinearly converged solution strategies.

Nonlinearly converged approaches attempt to advance the fully discrete model as a simultaneous inversion of coupled equations. For time-dependent problems, the discrete model is often implicit because of numerical stiffness (usually a result of the spatial discretization or fast time scales in the original problem). The nonlinear implicit system must be iterated to convergence. Done properly, nonlinear convergence ensures nonlinear consistency among physics components and is the "gold standard" for solving strongly coupled nonlinear systems.

However, the sheer size of target physics problems at the exascale, combined with the expected hardware limitations of exascale computing in terms of memory and resiliency, will severely restrict practical nonlinear solution strategies. For instance, in scale-bridging applications, nonlinear solution approaches attempting to converge on a nonlinear residual in which all variables of all levels of the hierarchy are listed explicitly will likely become prohibitive. Nevertheless, the benefits of nonlinear convergence in terms of accuracy and robustness suggest that nonlinear iterative approaches will continue to play an important role at the exascale (Section 4.4). In order to meet exascale needs, further research into nonlinear solvers will be required that limits global communication, that minimizes memory footprints, and that can take advantage of acceleration via preconditioning. One important focus of future nonlinear solver research should be enabling these algorithms to exploit the benefits of algorithmic partitioning (as described earlier in this section) in terms of memory frugality, modularity, task parallelism, and asynchrony.

Another important research topic for the exascale is practical preconditioners. The resources made available by exascale computers provide new directions for preconditioner research. For instance, when multiscale models are globally coupled, the less expensive coarse-grained model can be used as a preconditioner for the finer-scale problem. An example was demonstrated in the context of the hierarchy of models obtained through moment-based *model coarsening* as discussed in Section 4.2. The kinetic and moment models can be viewed as a "two-grid" multilevel approach, and the relatively inexpensive moment model calculations can be used to accelerate the convergence of the fine-scale solution using well-defined restriction (moment integrals) and well-posed prolongation (constraining kinetic descriptions by moment quantities) operators. An alternative strategy that leverages the additional computational power at the exascale may be to use a coarsely partitioned model with correct compatibility constraints (e.g., a partitioned approach to fluid-structure interactions) as a preconditioner for a better-resolved, fully coupled system.

Stability and consistency. Exascale computing will not change the fundamental tenet that discrete algorithms must be stable and consistent. Stability and consistency will continue to be essential for algorithms that couple nonlinearly disparate scales or physical models for exascale computing.

With regard to stability, one must go beyond linear stability analysis and consider nonlinear stability as well. Important elements in this regard are the nature of the numerical coupling (e.g., tight vs. loose), the preservation of conservation laws, the asymptotic well-posedness of the formulation, and the analysis and characterization of nonlinear stability through the use of nonlinear analysis tools such as modified equation analysis and variational formulations (when available). Many of these analysis techniques are related. For instance, variational formulations are intimately connected to the preservation of conserved quantities, and the latter provide constraints to ensure nonlinear stability.

With regard to consistency, the constraints of exascale computing will favor high-order, compute-intensive, and memory-frugal formulations. However, one must ensure not only that design order of accuracy is obtained for sufficiently small discretization parameters (e.g., time steps and mesh spacings) but also that the formulations are *asymptotic-preserving* [54] when these parameters get large with respect to some characteristic scale in the system (either temporal or spatial), as they will in scale-bridging applications. In particular, one must ensure that the proper physical asymptotic limit is achieved by the discretization of choice when time steps or mesh sizes do not resolve microscopic physical phenomena. Much work has been done recently in this area, in the context of both temporal and spatial discretizations. However, asymptotically well-posed discrete formulations often tend to feature low orders of accuracy in resolved scales. Thus, the development of high-order, asymptotically well-posed numerical formulations will be a key area of mathematical research in the exascale era.

Another sense of consistency is expected to play a role at the exascale: the discrete preservation of solution invariants (e.g., conservation laws), at both the local and the global levels. Locally, such invariants may be useful to detect soft faults. For instance, if one expects a local quantity to be conserved discretely and it is not after a given computation, the offending operation may be repeated. Globally, invariants limit the solution manifold and thus prevent solution drift over long simulations when exactly satisfied. This property is particularly important for conservation laws that apply across scales in scale-bridging algorithms. Discrete consistency with solution invariants must be designed into the discretization if these properties are to be leveraged to ensure correctness.

Particle-based approaches bring challenges in both stability and consistency [9, 20, 67]. Despite the success of some of these approaches in obtaining first-of-their-kind simulations, there is still much to be understood about the generation, propagation, and nonlinear interaction of errors due to interpolation and stochastic noise, as well as the impact of this noise on the overall accuracy of the simulation for long-time integration. Research on the development of low-noise techniques that treat collisions and remapping techniques that preserve conservation laws will be important components of the exascale mathematics portfolio.

Stability and consistency for stochastic systems are difficult questions in general. Numerical treatment of stochastic systems is not simply a matter of adding stochastic forcing to an existing method. In some cases, capturing the stochastic structure of the system places stringent demands on how deterministic terms are discretized. Numerical methods for stochastic systems retain all of the complexity of methods for deterministic problems and have additional requirements for capturing the probabilistic structure of the system. Although considerable research has been done in this area, numerical methods tend to lag behind their deterministic counterparts. A need exists for both richer classes of algorithms and a deeper understanding of convergence behavior for stochastic systems.

4.3.2 Parallel-in-Time Discretizations

Since clock speeds are no longer increasing, a significant challenge for the computational science community on future computer architectures is to overcome the sequential nature of current time integration methods. At first thought, this seems an impossible task. But development of parallel-in-time methods actually dates back almost 50 years [61], and significant speedups over traditional time marching approaches have already been demonstrated. Relative to standard time integration methods, however, the volume of research and development done in this area is extremely small and certainly not enough to move us into the exascale era. For most practitioners, the move to a parallel time integration setting will be a major paradigm shift that will have a huge impact on existing codes, the algorithms used, and even the way simulations are visualized and computationally steered.

One way to understand how parallel-in-time algorithms work is to consider the fully discretized space-time system. Traditionally, the system is solved by marching from one time step to the next, much as is done in a forward solve for a lower-triangular matrix. This approach is computationally optimal, but sequential. The idea is to instead solve the same space-time system by computing multiple time steps at once in an iterative fashion. If the iterative method is also computationally optimal and exhibits enough concurrency, then additional parallel resources can be used to achieve a speedup. Note that, in practice, it is not necessary to solve the full space-time system at once, just one space-time slab at a time.

There are many interpretations of parallel-in-time algorithms that may prove useful in developing efficient, robust formulations. For instance, methods such as parareal [58] can be viewed as classical two-level nonlinear multigrid algorithms (even though they were not originally introduced as such), and recent work has generalized this idea to fully multilevel methods in space and time. Since multilevel algorithms have been shown to be optimal (see Section 4.4.2) and have a high degree of concurrency, such parallel-in-time approaches are ideal, at least for large enough problems. Other perspectives treat the method as an extension of multiscale, multiphysics schemes in time or as a sophisticated nonlinear iteration for a space-time discretization. Some approaches to parallel-in-time attempt to be discretization agnostic, while others exploit specific discretization strategies such as the interesting work done on spectral deferred correction schemes and on first-order system least squares.

In all cases, one of the main research issues is achieving optimal convergence of the iteration, a major component of which is formulating an appropriate coarse-scale problem. Relatively straightforward approaches may work in some cases, while in other cases more elaborate ideas may be needed. For example, three ideas presented at the exascale workshop were the use of a least squares shadowing problem for solving chaotic systems [WP37], the use of the slow time-scale component of the underlying PDE for oscillatory systems [WP1], and the use of different governing equations on different levels [WP46]. In general, significant research remains to be done.

4.3.3 High-Order Discretizations

Many multiphysics applications in use today use low-order discretizations (first- or second-order). Higher-order discretizations have been less used for several reasons, but chief among them are numerical stability, difficulty of implementation, and perceived computational expense. By design, high-order methods have little implicit numerical dissipation to stabilize the scheme, particularly at boundaries, and so their implementation requires careful analysis and consideration. High-order discretizations also require more floating-point operations than do their low-order counterparts; thus, on the *same mesh*, high-order methods require more CPU time.

In the exascale realm, however, floating-point operations will effectively be free relative to the cost of data motion. Furthermore, high-order discretizations use fewer nodes, elements, or cells to achieve a required level of accuracy, provided that the features to be represented are well-resolved. This is just an inversion of the definition of a high-order discretization: higher-order methods asymptotically have a smaller error for a given mesh size. Of course, unless the discretization is compact, high-order methods have larger stencils or more degrees of freedom within each element, which can lead to additional data transfer and/or denser linear algebra problems. If properly managed, there is potentially less data motion for high-order methods than for low-order, which will have benefits for exascale performance. On node, high-order methods will have higher operational intensity than low-order methods, that is, will do more operations per byte of data loaded, and so will be better able to reach the performance limit of the node. In practice, the actual operational intensity is limited by the implementation details, so research is needed into structuring high-order algorithms to achieve, as much as possible, the theoretical operational intensity. In contrast, low-order schemes are typically bandwidth-limited, that is, because there is less computational work, the floating-point performance is limited by the data transfer speeds, and no amount of reimplementation will ever reach the maximum on-node performance.

In addition, the move to high-order methods raises further applied mathematics research questions that require research. Much analysis is needed to devise stable interior and boundary discretizations. Scalable, high-order temporal discretizations will be needed that avoid global synchronization. Splitting schemes (model partitioning) will need to be coupled in a high-order way, particularly at boundaries where the compatibility conditions must be respected. High-order methods lead to more coupled linear algebra problems (denser matrices) that will require suitable solvers. Moreover, high-order methods will require improvements in techniques to handle high-order computational geometry representations and meshes.

4.3.4 Adaptivity

Just as in petascale computing, adaptive mesh refinement (AMR [11]) will continue to be an important algorithmic component at exascale. When solution features are isolated in space, AMR reduces the amount of data and provides a natural partitioning of the simulation domain into regions requiring greatest resolution. For semi-structured AMR techniques, the underlying multilevel structure provides a natural and efficient hierarchy that should be favorable for scalable algorithms. Thus, AMR promotes concurrency and reduces memory usage. The availability of more concurrency should help reduce the overhead associated with refinement, regridding, and interlevel communications. Nevertheless, exascale introduces design constraints, such as limiting data motion and favoring compute-intensive kernels, that will encourage the consideration of complementary strategies such as local *order adaptivity*, which introduces different challenges in load balancing and asynchronous execution.

Model adaptivity, where the model changes locally in a region of the domain, will also play an important role in exascale computing. In fact, model adaptivity will be most effective when combined with adaptive mesh and algorithm refinement techniques [45]. These models can describe the same physics at different levels of fidelity at the same location (e.g., fine-grained models that replace constitutive relations) or can describe different physics altogether (e.g., fluid-structure interactions, multiphase simulations). Model adaptivity features several advantages for exascale computing: it can potentially exploit different levels of parallelism, asynchrony, and mixed precision and can minimize communication across layers. Model adaptivity (as well as AMR) could be tied to error control and uncertainty management to apply the finer-grained models only in those regions where the extra expense improves the solution accuracy. Model adaptivity still presents important challenges, however, particularly with regard to the nature of the coupling across different physical descriptions, the criteria governing the choice of model, dynamic load balancing, the preservation of conservation laws, and the impact on fidelity of the prolongation of information from coarser descriptions to finer ones. There are clearly opportunities in developing scalable adaptive algorithms, but more research is needed.

Real-world simulations can also transition between regimes of coupling strength, and this feature will likely be more pronounced in exascale multiphysics simulations. Detecting these transitions and devising *adaptive discretizations* that minimize error relative to fully coupled models could provide substantial savings by using more sophisticated (and expensive) coupling procedures only when necessary. Using metrics that determine the strength of coupling between operators in multiphysics models, dynamically adaptive discretizations could adjust solution approach (e.g., explicit or implicit) per component, the coupling accuracy, and/or the splitting method during the course of long-running simulations. Research will be required to ensure a consistent solution from such algorithms as well as to develop verification approaches that accommodate the solution- and time-dependent discretization.

4.3.5 Scalable Computational Geometry and Mesh Generation

To date, a gap has remained in the exascale discussion regarding issues related to defining and constructing complete simulation workflows. The execution of a simulation workflow begins with the domain geometry and includes many aspects, from meshing and discretization to solvers, error estimation, adaptive refinement, data transfer between meshes, UQ, optimization, and visualization. The interactions among these components are complex, can be tightly coupled, and occur throughout the entire solution process. Thus, when executed on massively parallel computers, parallel structures and services are required for all aspects of the simulation workflow. Moreover, different workflow components are often developed by different work groups. In order to allow simulations to leverage the best available tools in each category, it is important that the parallel structures be based on well-described functional interfaces that form the backbone of an interoperable infrastructure.

The two key technical issues discussed in this section are parallel methods for interacting with high-level descriptions of complex geometries and generating and adapting high-quality, high-order, curvilinear meshes. Both these workflow steps are an integral part of the solution process and must be done by using in-memory linkages among geometry, mesh, simulation fields, and adaptive control. Without additional research and development, these areas will be a critical bottleneck in next-generation DOE science applications that require high-order methods to solve multimodel problems over complex, high-dimensional domains.

Computational geometry. Historically, the simulation domain was defined directly in terms of the discretization (e.g., the computational mesh) used by the analysis procedures. Unfortunately, such a definition does not adequately support the automation needed in multimodel and/or adaptive simulations. A higher-level domain representation is required that can support the complete specification of the simulation attributes and all the geometry operations required by the simulation components. These operations include automatic generation and adaptation of meshes to the true geometry, geometry interrogations for analysis, and geometry modifications to account for large deformations, fracture, shape optimization, and adaptive control of geometric simplifications.

Clearly, the geometric operations described above require that the geometric information be available throughout the simulation process. However, when the geometric model contains large numbers of boundary entities or extensive shape information, maintaining a copy of the entire geometry on each process becomes problematic. It is not unusual for a moderately complex CAD model to be 1 GB of data, which is a significant percentage of the expected size of local memory in future computer systems, and therefore a possibly unacceptable storage overhead. In addition, the I/O associated with copying this amount of data to each process creates a bottleneck on massively parallel machines. In cases when the geometry input has millions of features (e.g., full models of integrated circuit), it is also important to generate the original, complete geometric model in parallel for the basic design data (e.g., GDSII layout files for integrated circuits). The development of parallel geometric modeling has received little consideration, but one approach that appears promising is a flexible, parallel spatial decomposition of the domain based on an octree for which parallel implementations have been developed for mesh generation.

When a parallel mesh has already been constructed for a given geometric model, one can distribute the model based on the distribution of the mesh, colocating the geometric model entities with their associated the mesh entities. The key to proper parallel operation is maintaining proper model entity adjacencies. As an example, consider the partitioned mesh and geometry shown in Figure 3. The upper image is the partitioned mesh, while the lower image shows the mesh on four of the sixteen parts and the geometric model entities (in gray) that are stored with those meshes. Tests show that the total memory to store the model entities in a distributed fashion is generally independent of the number of partitions and only up to 1.5 to 2 times more than storing the entire model a single time.

Additional research is required in a number of areas including understanding how to best partition a geometrical domain without the pre-existence of an associated parallel mesh, how to handle evolving geometry (e.g., in shape optimization and damage modeling); geometrical coarsening/refinement operations to match the level of accuracy required by the simulation; lossy and lossless compression schemes for geometrical information; and efficient, fast migration of geometric

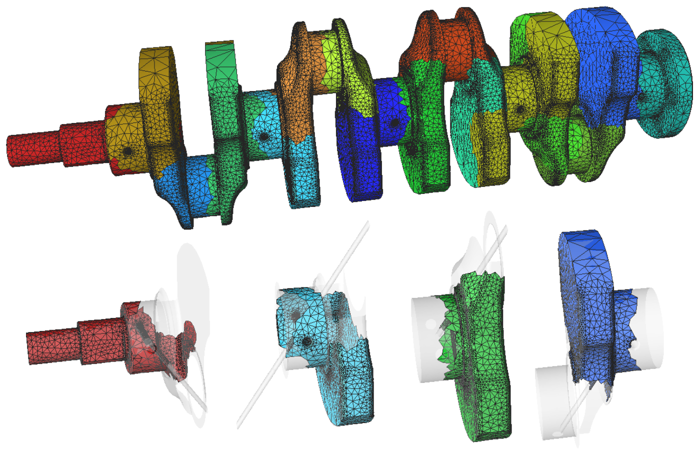


Figure 3: Parallel geometry for a distributed mesh. Geometry included with each part is shown as translucent.

information to different processors as the simulation proceeds.

Parallel mesh generation and adaptation. The majority of parallel simulations rely on an initial serial mesh, which introduces a bottleneck when the meshes have many millions of elements. The bottleneck is due to the cost of file transfer from the large-memory machines (typically used to generate the mesh) to the parallel computers used to execute the simulation. Alternatively, the initial meshes should be generated and adapted, in parallel, on the same parallel computer that executes the analysis. Methods to support distributed mesh generation and adaptation have been developed and are available for many types of meshes; but as meshes reach very large sizes, such methods become scarce. The bulk of this work supports MPI-based programming models, although current research efforts are increasingly investigating methods and programming models for hybrid parallelism and mixed GPU/CPU support.

Additional research is needed in a number of areas to ensure that this critical component of the simulation workflow supports the needs of high-order simulations and can run efficiently on exascale computers. While high-order discretization methods have been developed and studied extensively, the overall solution approaches have often used low-order (e.g., linear) representations of the computational mesh. These can limit the accuracy and convergence rates achieved by the simulations. Many fundamental, open questions remain that are related to the generation, quality control, and adaptation of high-order, curvilinear meshes, both in serial and in parallel. For example, new methods are needed for high-order mesh transformations (e.g., mesh smoothing or swapping) and dealing with moving meshes that contain high-order elements.

For the complex multiphysics applications that will be enabled by exascale computers, new methods of mesh generation and data partitioning are required. Dynamic partitioning methods for mesh adaptation must themselves be fast and scalable and must directly consider the needs of each simulation workflow step. It is often unclear what the partitioning objective function should be to maximize performance, and trade-offs among load balancing, maximizing memory bandwidth, and minimizing communication costs must be considered. Furthermore, since exascale computers will often be used for ensemble simulations (e.g., for uncertainty quantification or design studies) those scenarios must leverage information across as many runs as possible. For mesh and geometry information, it is unclear, for example, which information can be leveraged, what optimizations can be made, and whether any memory compression schemes can be achieved.

4.3.6 Related Position Papers

Many position papers related to discretization were presented at the Exascale Mathematics Workshop. These included papers on multiscale and multiphysics coupling [WP14, WP25, WP27, WP28, WP40, WP73], high-order discretizations [WP17], time discretization [WP53], and computational geometry [WP19]. Additional position papers related to these topics include [WP2, WP33, WP44, WP60, WP61].

4.4 Scalable Solvers

In our top-down view of the role of applied mathematics in exascale computing, discussion has moved from the specific (i.e., formulations and models specific to the problem at hand) to the more general. With an appropriate mathematical model chosen and discretized, the original problem is approximated (in general) by a finite-dimensional, coupled, nonlinear or linear system of algebraic equations. Nonlinear solvers, eigensolvers, and linear solvers, appropriate to the properties of the algebraic system, are then employed to obtain the approximate solution. Many of these solvers are provided in numerical solver libraries, which have been optimized over the years for a variety of platforms.

Moving to the exascale will put heavier demands on these algorithms in at least two areas: the need for increasing amounts of data locality in order to perform computations efficiently and the need to obtain much higher factors of fine-grained parallelism as high-end systems support increasing numbers of compute threads. Consequently, parallel algorithms must adapt to this environment, and new algorithms and implementations must be developed to capitalize on the computational capabilities of the new hardware. Here, we discuss several key research topics in numerical solver development for exascale computing. While the discussion focuses mostly on the solution of linear systems, many of the issues and challenges identified are applicable to nonlinear solvers and eigensolvers.

4.4.1 Direct and Iterative Solvers

The solution of sparse linear systems is often the most time-consuming computation in large-scale science and engineering simulations. Although iterative methods have become more and more prevalent for solving many of these systems, direct solvers are still widely used and will continue to play an important role at the exascale. The primary advantage of direct solvers is their robustness. Direct solvers are guaranteed to terminate after a finite number of steps. Moreover, with appropriate pivoting for numerical stability, direct solvers are often the method of choice for solving highly ill-conditioned linear systems. The main drawbacks of direct methods are their memory requirement and computational cost. Pinpointing the complexity of direct methods is difficult because of fill (which refers to the zero entries of the matrix that become nonzero during the factorization process). For matrices arising from certain two-dimensional finite-element/finite-difference meshes, the number of nonzero entries in the triangular factors and the number of operations required to compute the factors are bounded below by $O(n \log n)$ and $O(n^{3/2})$, respectively, where n is the number of unknowns. In three dimensions, these numbers are much larger. In general, the complexities are significant for large problems compared with iterative methods, particularly $O(n)$ solvers such as multigrid (discussed below).

For iterative methods, the primary kernel in an iteration is typically a matrix-vector multiplication, the cost of which is generally proportional to the number of nonzeros in the matrix. On the other hand, the convergence of iterative methods can vary widely across applications and even within a single code or simulation. The number of iterations required depends on the linear system being solved. The iteration count can be affected by applying preconditioning to the linear systems, although the choice of preconditioners is often application dependent. Attempts have been made to employ techniques developed for sparse direct solvers to compute incomplete factorizations, which are then used as preconditioners for iterative methods. Such approaches have varying degrees of success.

In addition, effective iterative methods often have subcomponents that require direct solvers, both dense and sparse. For example, dense solvers are used to build smoothers in multigrid methods for PDE systems, and sparse direct solvers are sometimes needed to solve the coarse system of equations. In domain decomposition, a linear system is permuted so that it has a bordered block diagonal form. Direct solvers (and iterative solvers too) are ideal for solving the diagonal blocks because these blocks are smaller. Processing the border results in the Schur complement, which can be dense, and can be solved by using iterative methods, since the Schur complement needs not be formed explicitly. Several implementations of such hybrid methods have been proposed and applied to large-scale problems. In other variants, direct and iterative methods are combined to produce efficient hybrid solvers.

In general, dense and sparse linear algebra represent fundamental building blocks that are ubiquitous and used in a variety of applications. Although they will be rarely used on a full exascale system, dense operations frequently occur on smaller scales ranging from a single multicore processor to accelerators and terascale/petascale clusters of such components. For all these reasons, research and development in both direct and iterative solvers will be essential for future exascale simulation science needs.

4.4.2 Multilevel Algorithms

For many problems, the fastest and most scalable solver approaches are multilevel methods, because they are both mathematically optimal and highly parallel. As a result, multilevel solvers are already widely used in DOE scientific simulation codes, and we argue that their importance only increases in the exascale setting. Consider the simplest setting of multigrid methods for linear systems (the basic comments and conclusions carry over to the general setting). Multigrid methods are called optimal (order) methods because the work required to solve a linear system is linearly proportional to the number of unknowns. That is, they are $O(n)$ methods, where n is the number of unknowns. This property gives them the potential to solve ever larger problems on larger parallel machines in (nearly) constant time. Multigrid methods achieve this optimality by employing two complementary processes: smoothing and coarse-grid correction. In the classical setting [7, 13, 43] of scalar elliptic problems, the smoother (or relaxation method) is a simple iterative algorithm such as Gauss-Seidel that is effective at reducing high-frequency error. The remaining low-frequency error is then accurately represented and efficiently eliminated on coarser grids via the coarse-grid correction step. Applying this simple multigrid idea to get a scalable method often involves considerable algorithmic research, however. One has to decide which iterative method to use as a smoother, how to coarsen the problem, and how to transfer information between the grids. When designed properly, a multigrid solver is algorithmically scalable; it uniformly damps all error frequencies with a computational cost that depends only linearly on the problem size. In addition, a well-designed multigrid algorithm has a high degree of concurrency. Specifically, its computational task dependency graph has a depth that depends at most logarithmically on the problem size. In other words, the size of the sequential component of the algorithm (the part that cannot be parallelized) is only $O(\log n)$, which is often the minimum-order size achievable because of the underlying physics being simulated.

Because of their optimality and high concurrency properties, multilevel methods will continue to play a critical role for exascale computing. In addition, at least one parallel algebraic multigrid code has already been shown to exhibit a natural resilience to soft faults when applied to diffusion problems, with the vast majority of solver failures being due to pointer corruption and not mathematical frailty. This result may indicate that multilevel methods are a good starting point for building fault-tolerant algorithms as well. Finally, multilevel techniques will also likely play an important role for computing multiple time steps in parallel, as discussed in Section 4.3.2.

Although multilevel methods have many desirable properties, any given algorithm generally has somewhat narrow applicability. As a result, there are many approaches for solving different classes of problems, and in some cases optimal-order methods have yet to be fully developed (for example, Helmholtz equations). In addition, exascale computing restricts algorithmic choices, eliminating the use of important techniques such as lexicographic Gauss-Seidel smoothing (too sequential) and W-cycles (too much communication). Hence, a high-level multigrid research agenda basically involves the development of optimal algorithms that address at least one or more of the following: (1) new application areas; (2) removal of sequential subcomponents (e.g., smoothers that follow characteristics in CFD applications); or (3) communication reduction (e.g., additive methods, non-Galerkin coarse operators, multilevel domain decomposition approaches). Another important research goal is the pursuit of methods that are broadly applicable. Algebraic multigrid (AMG) [14, 64] is an example of such a research topic that has paid dividends to date. In practice, AMG is tailored for specific applications to achieve the best performance, usually by way of many adjustable parameter choices. But, the basic AMG goal of developing an $O(n)$ method that depends only on the coefficients of a general matrix has helped further our overall knowledge and understanding of multilevel methods and has led to breakthroughs in areas such as lattice quantum chromodynamics, where the development of optimal multigrid methods had been illusive.

4.4.3 Numerical Solver Exascale Research Issues

While the anticipated changes in architecture, as discussed in Section 3, will have effects across the "math stack," it is in numerical solvers that the necessary adaptations may be most explicit. Here we discuss several strategies and techniques that should help achieve high performance but that require further investigation: communication-avoiding algorithms; synchronization reduction; data compression; mixed-precision algorithms; randomization and sampling algorithms; adaptive load balancing; scheduling and memory management for heterogeneity; energy-efficient algorithms; and autotuning. These ideas will also be useful in considerations of problem formulation, modeling, and discretization, since the requirements driven by the science needs will inevitably be tempered by the constraints of the computer architecture.

Communication avoiding. Algorithmic complexity is usually expressed in terms of the number of operations performed rather than the quantity of data movement to memory. This is antithetical to the expected costs of computation at the exascale, where memory movement will be very expensive and operations will be nearly free. When solving very large problems on parallel architectures, the most significant concern becomes the cost per iteration of the method—typically because of communication and synchronization overheads. This is especially the case for preconditioned Krylov methods, for example, which are the most popular class of iterative methods for large sparse systems.

To address the critical issue of communication costs, researchers need to investigate algorithms that minimize communication. New bandwidth and latency lower bounds must be derived for various numerical algorithms on parallel and sequential machines (e.g., for dense linear algebra algorithms where the well-known lower bounds for the usual $O(n^3)$ matrix multiplication algorithm should be extended). New algorithms that attain these lower bounds, at least in many cases, must be invented. Another example of needed research is in Krylov subspace methods such as GMRES, CG, and Lanczos, where one should devise means to take k steps of these methods with the same communication costs as a single step.

One method that should be exploited more is the fast multipole method (FMM) algorithm, which exploits local regularity to achieve significant reductions in both computation (asymptotically, from $O(n^2)$ to $O(n)$) and communication. This algorithm has been named one of the most important algorithms of the 20th and 21st centuries, yet strangely it has seen little adoption, perhaps because of its initial perceived complexity. However, a growing community of algorithms experts are finding that the future success of the exascale era may go hand in hand with the ability for researchers to develop new, fast direct solvers and adaptations to the FMM. The route to this may involve using kernel-independent approaches that remove the need for FMM solvers to be hand-crafted for particular applications. The breadth of application of FMM is not limited to such solvers; one important application will be its ability to reduce communications of large FFTs by close to a factor of 3.

Synchronization reduction. Often one must synchronize the computation in an algorithm. A good example is the parallel computation of dot products. Synchronization is needed after such global reductions. However, synchronizations can become bottlenecks. Thus, one must design algorithms that have as few synchronization points as possible. Attempts have been made to restructure existing algorithms so that the number of synchronizations is reduced. An example is the conjugate gradient algorithm. By using some mathematical identities, one can produce versions of the conjugate gradient algorithm that have just one synchronization rather than two in the conventional description of the algorithm.

The idea of restructuring the algorithm to reduce the number of synchronizations, and in general the amount of communication, will become more important in the exascale era. However, not all the variants of an algorithm may have the same numerical behavior. In the case of conjugate gradient, some variants may not be numerically stable. Thus, in restructuring an algorithm to reduce synchronization and communication, the stability of the variants is an important consideration.

Data compression. Another way to reduce the volume of communication is to consider data compression. If the compression rate is high, then this can result in a significant reduction in the amount of data to be communicated in an algorithm. In some cases, data compression can also result in an improvement in the execution time despite the fact that time is needed to perform the compression.

An example of data compression is the recent work on matrix factorizations using compact representation. For matrices arising from self-adjoint elliptic operators, for example, submatrices in the factors exhibit low ranks. Thus, one can store these low-rank submatrices by using compact representations, such as singular value decompositions, rather than the conventional matrix representation. Results have shown significant reduction in the storage requirement, and naturally this also leads to reduction in the amount of data to be communicated in a parallel setting. Furthermore, the cost of the factorization is often reduced because less data has to be manipulated, even though time is needed to compute the compact representations.

The idea of low-rank representations can be extended further by allowing lossy compression. Using the matrix factorization as an example again, one can truncate the singular value decomposition to obtain a low-rank approximation. This results in an approximate factorization, which can then be used, for example, as a preconditioner in an iterative method.

A number of research problems should be investigated. First, what is the extent of compression possible for a given class of problems? Second, what is the tradeoff between the increased cost due to compression and the possible reduction in cost due to reduction in communication? Third, when lossy compression is enabled, what is the impact on the reliability and accuracy of the algorithm?

Multiple-precision algorithms. Algorithms and applications are becoming increasingly adaptive, and we have seen that various adaptivity requirements have become an essential, key component of their roadmap to exascale computing. Another aspect of this quest to adaptivity is related to the development of libraries that recognize and exploit the presence of mixed-precision mathematics. A motivation comes from the fact that, on modern architectures, the performance of 32-bit operations is often at least twice as fast as the performance of 64-bit operations. Moreover, by using a combination of 32-bit and 64-bit floating-point arithmetic, the performance of many linear algebra algorithms can be significantly enhanced while maintaining the 64-bit accuracy of the resulting solution. This approach can be applied not only to conventional processors but also to other technologies, such as GPUs, and thus can spur the creation of mixed-precision algorithms that more effectively utilize heterogeneous hardware.

Mixed-precision algorithms can easily provide substantial speedup with little coding effort mainly by taking into account existing hardware properties. Earlier work has shown how to derive mixed-precision versions for various architectures and for a variety of algorithms for solving general sparse or dense linear systems of equations. Typically, a direct method is first applied in single precision in order to achieve a significant speedup compared with using double precision. Then an iterative refinement procedure aims at retrieving the lost digits. Iterative refinement can also be applied for eigenvalue and singular value computations.

Of current interest is to extend and incorporate this approach in applications that do not necessarily originate from linear algebra and to study the robustness of mixed-precision algorithms on large-scale platforms. Indeed, the convergence of the mixed-precision iterative refinement solvers strongly depends on the condition number of the matrix at hand. The conditioning can be determined at run time, and proper precision can be selected. Ideally, the user could specify the required precision for the result, and the algorithm would choose the best combination of precision on the local hardware in order to achieve it. The actual mechanics would be hidden from the user.

Randomization and sampling algorithms. Randomized and asynchronous algorithms have been successful in several areas of computer science and have received a growing amount of interest in recent years in linear algebra, in particular linear least squares (dense) and general sparse linear systems. On future exascale systems, randomization algorithms and algorithms based on sampling may become more important in reducing synchronization and data movement. This approach may outperform deterministic algorithms for standard problems such as solving linear systems. Randomized algorithms have the advantage that they are often simple to implement and often require little synchronization; some versions may run completely asynchronously. However, randomized algorithms often raise concerns. Are they sufficiently accurate? Do they converge too slowly for a given tolerance? What if they fail? Success "with high probability" is not acceptable in many cases. One place where randomized algorithms may play an important role is as preconditioners in iterative methods. For preconditioners, which try to accelerate convergence, it is acceptable to have low accuracy and occasionally fail to return a correct answer, since there is an outer iteration to guarantee convergence.

Adaptive response to load imbalance. As we move to architectures with billions of threads, even naturally load-balanced algorithms on homogeneous hardware will present many of the same load-balancing problems that are observed in current adaptive codes. For example, software-based recovery mechanisms for fault tolerance or energy-management features will create substantial load imbalances as tasks are delayed by rollback to a previous state or correction of detected errors. Dynamic scheduling based on directed acyclic graphs (DAGs) has been identified as a path forward, but this approach will require new approaches to optimize for resource utilization without compromising spatial locality.

Scheduling and memory management for heterogeneity and scale. Extracting the desired performance from environments that offer massive parallelism, especially where additional constraints (e.g., limits on memory bandwidth and energy) are in play, requires more sophisticated scheduling and memory management techniques than have heretofore been applied to linear algebra libraries. Confronting the limits of domain-decomposition in the face of massive, explicit parallelism introduces another form of heterogeneity. Feed-forward pipeline parallelism can be used to extract additional parallelism without forcing additional domain-decomposition, but it exposes the user to dataflow hazards. Ideas relating to a data-flow-like model, where parallelism is expressed explicitly in DAGs, allows for dynamic scheduling of tasks, support of massive parallelism, and application of common optimization techniques to increase throughput. Approaches for isolating side-effects include explicit approaches that annotate the input arguments to explicitly identify their scope of reference and implicit methods, such as using language semantics or strongly typed elements to render code easier to analyze for side-effects by compiler technology. New primitives for memory management techniques are needed that enable diverse memory management systems to be managed efficiently and in coordination with the execution schedule.

Energy-efficient algorithms. Emerging constraints on energy consumption are expected to have pervasive effects on HPC; power and energy consumption must now be added to the traditional goals of algorithm design, namely, correctness and performance. The emerging metric of merit is performance per watt. Consequently, it may be essential to build power and energy awareness, control, and efficiency into the foundations of our numerical libraries. In order to accomplish such a goal, first and foremost, standardized interfaces and APIs for collecting energy consumption data should be developed, just as PAPI has done for hardware performance counter data. Accurate and fine-grained measurement of power consumption underpins all tools that seek to improve such metrics; anything that cannot be measured cannot be improved. Second, these tools must be used to better understand the effects that energy saving hardware features have on the performance of linear algebra codes. Third, parameters and alternative execution strategies must be identified for each numerical library that can be tuned for energy efficient executions, and to enhance schedulers for better low-energy execution.

Autotuning algorithms. Numerical algorithms and libraries need the ability to adapt to the possibly heterogeneous environment in which they operate. Such adaptation must deal with the complexity of discovering and applying the best algorithm for diverse and rapidly evolving architectures. An automated process would be best, both for productivity and for correctness, where productivity refers both to the development time of the implementation and to the user's time to solution. The objective is to provide a consistent library interface that, independent of scale and processor heterogeneity, can achieve good performance and efficiency by binding to different underlying code, depending on the configuration. The diversity and rapid evolution of today's platforms mean that autotuning of libraries such as the BLAS will be indispensable to achieving good performance, energy efficiency, and load balancing across the range of systems. In addition, autotuning has to be extended to frameworks that go beyond libraries, such as optimizing data layout (e.g., blocking strategies for sparse matrix/SpMV kernels), stencil autotuners (since stencils kernels are diverse and not amenable to library calls), and even tuning of the optimization strategy for multigrid solvers (optimizing the transition between the multigrid coarsening cycle and bottom-solver to minimize runtime). Adding heuristic search techniques and combining these with traditional compiler techniques will enhance the ability to address generic problems extending beyond linear algebra.

4.4.4 Role of Numerical Libraries

Many numerical solvers for high-performance computing are made available through libraries, and DOE has historically supported the development of such libraries. LAPACK, ScaLAPACK, PETSc, hypre, Trilinos, SuperLU, SUNDIALS, Chombo, BoxLib, SAMRAI, and TAO are well-known examples of the results of this investment. Numerical libraries will continue to play an important role at the exascale; once genuinely exascale-suitable algorithms for a class of discrete problem have been identified and developed for a particular platform, libraries provide an efficient means to share these implementations across applications with similar characteristics. Of course, libraries alone cannot provide all the advances needed for exascale. Expert knowledge on the algorithms and how to use them effectively within the context of the problem, mathematical model, and discretization will continue to be important.

A critical issue for exascale computing, therefore, will be to develop numerical libraries for exascale architectures in order to share this wealth of experience efficiently, but this library development will face challenges. For instance, programming models and hardware architectures are still in a state of flux, and this uncertainty will slow down the development of extreme-scale solver libraries as new configurations and abstractions are tried. It seems most reasonable to build on top of existing libraries instead of developing entirely new libraries; this will amortize some of the software maintenance costs, provide backward capability, and make transition for applications easier. Many applications will need to be run on at least capacity up through leadership-class machines, if not down to even smaller-scale clusters and workstations. Libraries that can handle all of these scales of computing with consistent interfaces will aid in the development and portability of DOE application codes; autotuning is an obvious approach to pursue. Finally, the development of exascale-suitable extensions of numerical libraries will require more than just research into improved algorithms—it will also require significant investment into substantial software development and support that cannot (and should not) be a hidden cost of discrete solvers research.

4.4.5 Related Position Papers

Many position papers related to discrete solvers were presented at the Exascale Mathematics Workshop. Topics of presented papers included multilevel algorithms [WP9, WP16, WP17, WP20, WP21, WPA1], direct/iterative methods [WP34, WP45, WP56, WP59, WP65], eigensolvers [WP18], the use of compression techniques [WP45, WP47, WP65], communication/synchronization avoiding [WP3, WP22, WP68], randomization and sampling [WP47, WP59], partitioning and load balancing [WP43, WP44], and the use of mixed-/adaptive-precision arithmetic [WP25, WP54, WP65]. Additional positions papers related to this topic include [WP11, WP63].

4.5 Data Analysis

Undeniably, the exascale era will usher in unprecedented volumes of scientific data, including data captured at experimental facilities and data generated at leadership computing facilities. However, without efficient and effective methods for data analysis, the scientific advances—whether planned or fortuitous—buried in this data will be delayed or remain undiscovered.

Data analysis cuts across and/or has implications for all branches of the exascale mathematics stack detailed in this report. Computer simulations can be used to quantify the uncertainties in complex physical systems, and physical experiments can be used to validate the computer simulations. For example: Section 4.2.2 discusses advances in UQ that could enable scalable data fusion, and using experimental data and simulation to quantify the uncertainties in QoIs; developments in optimal experimental design (see Section 4.2.3) could change the data we capture and inform which experiments should be run physically and which should be run computationally; Section 4.4.3 underscored opportunities for data compression to reduce time to solution, while this discussion focused on numerical solvers, the fundamental questions are the same for more general analysis paradigms. Furthermore, many data analysis problems can be posed as problems in UQ or optimization or determined as solutions to differential and/or algebraic equations. In the remainder of this section we highlight opportunities for data analysis not covered in the rest of the report.

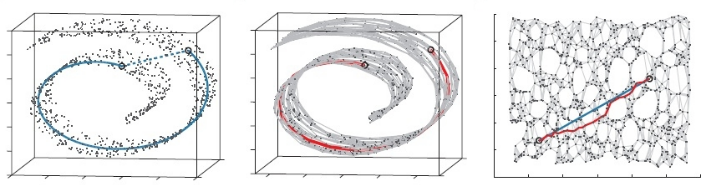
4.5.1 Concurrent Analysis

The traditional workflow for many data-intensive tasks arising from physical or computational experiments is that analysis is done offline, typically as part of a post-processing step. Steady improvements in both physical detectors and computing resources are enabling ever more enhanced experiments, but I/O constraints are already impeding the impacts of these improvements. For example, despite increases in temporal resolution, the gap between time steps saved to disk keeps increasing. This compromise in fidelity makes it impossible to track features with timescales smaller than that of I/O frequency. Such discrepancies will become more pressing on future architectures as increases in computational power significantly outpace I/O capabilities and will motivate a fundamental shift away from postprocess-centric data analyses.

Concurrent analysis frameworks are a promising direction wherein raw data is processed as it is generated, decoupling the analysis from I/O. Both in situ analysis frameworks, where operations share the primary resources used in data generation, and in-transit analysis frameworks, involving asynchronous data transfers to secondary resources, store only the results, which are typically several orders of magnitude smaller than the raw data. This reduction mitigates the effects of limited disk bandwidth and capacity. Since these solutions involve the sharing of resources, they face significant challenges because analysis and simulation algorithms must be redesigned to operate within tight memory, communication, and I/O constraints. A further challenge is performing several analyses simultaneously, within these constraints and in a coordinated fashion.

4.5.2 In Situ Data Reduction and Transformation Techniques

A shortcoming of concurrent analysis frameworks is that they require a priori knowledge of the questions one wants to ask of the data, thus limiting one to the study of anticipated phenomena. In many cases, unexpected results lead to new questions, which call for iterative exploration that may be most effectively done by postprocessing. Traditional compression techniques reduce the amount of data written to disk but, since the data must be decompressed prior to analysis, ultimately still require scalable analysis algorithms.


(a)

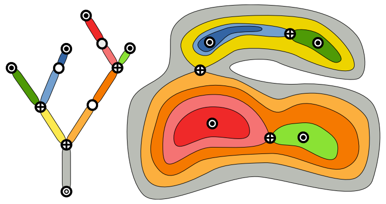

(b)

Figure 4: (a) Isomap uses geodesic distances on a weighted graph to identify a lower-dimensional embedding of high-dimensional data (image courtesy of [8]); (b) Merge trees segment data according to the level-set behavior of a field of interest (image courtesy of [10]).

An alternative solution is the use of in situ data transformations that create reduced representations while maintaining the properties of interest, thus minimizing the impact on subsequent analyses. Statistical feature extraction is one possible approach to identify a reduced representation of data. Such techniques—including dimensionality reduction algorithms such as principal component analysis and its variants, Isomap, and locally linear embeddings—define a lower-dimensional representation that still captures the data with sufficient accuracy (see Figure 4a). In contrast, segmentation-based feature extraction techniques focus on the identification of relevant subsets of a spatial domain. Typically, the subsets of the domain are defined in terms of one or more of the observables in the spatial fields and the resulting coherent structures correspond to physical phenomena of interest. Segmentation algorithms include those commonly used in the image analysis community (e.g., as used in analyzing medical scans) and topology-based, multiscale algorithms based on level-set or gradient behavior of a function defined on a spatial domain (see Figure 4b).

4.5.3 Memory- and Compute-Efficient Algorithms

Whether because of the need to share computational resources with a data-generating process or because of the sheer volume of required data, data analysis algorithms will need to operate under tight memory, communication, and I/O constraints. Not all analysis algorithms scale in such an environment, and significant algorithmic shifts will be required to achieve the necessary scalability.

We expect sampling-based algorithms to play an important role in this area. Sublinear algorithms are particularly interesting, since they are designed to estimate properties of a given function over a massive discrete domain, by accessing a tiny fraction of the domain. Sublinear algorithms have been used in graph analysis and the application-independent generation of colormaps; understanding in what settings these algorithms admit quantification of the error due to sampling is a key research question. Another class of memory-efficient algorithms that require further mathematical analysis is streaming techniques, which progressively process and visualize large scientific datasets by leveraging progressive multiresolution data structures.

Efficient implementation of data analysis algorithms on today's high-performance computing platforms often requires detailed knowledge of the network, memory hierarchies, and computing architecture. The extreme concurrency and resilience requirements of exascale computing create an even more pressing need for abstractions and frameworks to support the development of new data analysis algorithms. Effective paradigms will support decomposition of the analysis. Candidates include new constructs to operate on a fine granularity of data, building blocks for portable and efficient parallel analysis algorithms, system-tailored kernel functions, and MapReduce-style problem decomposition.

4.5.4 Related Position Papers

Position papers related to data analysis that were presented at the Exascale Mathematics Workshop include [WP43, WP55, WP62]. Additional positions papers related to this topic include [WP34, WP48, WP49].

4.6 Resilience and Correctness

Computing an incorrect answer quickly is of no use to a scientist. Yet computing with exascale hardware poses several challenges in assessing and assuring the correctness of numerical simulation results. Resilience to faults has been identified as a critical need for future HPC systems [57]; the thousandfold increase in computational capabilities expected over the next decade, along with incorporation of techniques for reducing energy consumption, is predicted to increase the error rate of the largest systems. DOE has several critical mission deliverables, including annual stockpile certification and safety assurance for the NNSA and future energy generation technologies for the Office of Science. Computer simulations are key to meeting these deliverables and must be resilient enough to complete in time and correctly, in order to meet the respective critical mission need. In many cases, these simulations can take days, weeks, or even months to complete, which increases the computation's exposure to faults.

Both hard and soft faults are expected to occur with much greater frequency than on previous hardware. Uncorrected soft faults have the potential to corrupt computed solutions. Hard faults will need to be handled on the fly; halting and restarting an entire application because of the loss of a node, for instance, will be prohibitively expensive at the exascale. Dynamically recovering from either type of fault will introduce nondeterministic variability in resource usage, as will dynamic scheduling of tasks. Because of the nonassociativity of floating-point arithmetic, such nondeterminism will make bitwise reproducibility difficult at best and will complicate code correctness testing procedures, including code verification, where reproducible execution behavior is assumed.

Preventing all faults during exascale simulation will be impossible, and nondeterministic execution is likewise unavoidable without potentially severe performance penalties. Fault management will require developments in hardware, programming environments, runtime systems, and programming models; but mathematics will play an important role as well. The issue of correctness is ultimately a mathematical one and will require mathematics-informed solutions. Research will be required in order to devise efficient application-level fault-tolerance mechanisms and new procedures to verify code correctness at scale.

4.6.1 Resilient Algorithms

Mathematical algorithms have typically been designed under the assumption that the computer system is a reliable digital machine, although lack of floating-point arithmetic associativity has been a regular concern. Computer system faults certainly occur but have typically been handled by a checkpoint/restart (CPR) mechanism that lives outside the scope of algorithmic concerns.

Even so, a large body of work, commonly referred to as algorithm-based fault tolerance (ABFT), is concerned with detecting and correcting floating-point error by means of knowledge about the algorithm and use of metadata, or reconstituting lost state via the same mechanisms. A seminal paper in this area is [53].

Currently, we expect that the frequency of failures, size of data, and cost of checkpointing and restarting will lead us to further consider models and algorithms for resilience beyond CPR. Applications and the numerical engines that drive them will need to take a more active role in the detection or recovery from errors, or both.

In addition to classical ABFT approaches, we need to develop, as part of the design of resilient algorithms, computing models that support expression and execution of these algorithms. Beyond CPR, models that have emerged include the following:

- **Skeptical Programming**
  If we no longer assume that our computing systems are reliable digital machines, one approach to mitigating the impact of failure is to be "skeptical" of results that are produced by introducing simple, inexpensive validation tests. Often these tests can be derived from metaknowledge about the problem being solved, such as a conservation principle, orthogonality property, or valid range specification. Although skeptical programming cannot detect or correct all faults, it can help reduce the number of faults. Furthermore, some faults may only slow progress to solution and can be tolerated instead of halting execution.

- **Relaxed Bulk-Synchronous Programming (RBSP)**
  One of the first impacts of reduced reliability is performance variability. As low-level system failure rates increase, error detection and correction happen more frequently in the hardware and system software layers. These events preserve the reliable digital machine model but introduce variability in execution time. Many scalable applications are designed under the implicit assumption that equal work implies equal execution time, so that if the work of a parallel application is balanced, then the application should scale well on a parallel computer even if processors must be synchronized across during execution. Performance variability, when coupled with frequent collective operations, leads to severe limitations in scalability, especially as one reaches a million or more processes.

  With the introduction of MPI-3 [59], asynchronous neighborhood and global collectives now enable a "relaxed" bulk-synchronous programming model (RBSP). Given RBSP capabilities, one can now develop algorithms that potentially hide latency. Data from some applications (e.g., [WP39]) show measured variability in real settings.

- **Local Failure, Local Recovery**
  For parallel applications based on MPI, the current approach to dealing with the loss of a single process is to kill all remaining processes and restart the application. Since computational runs now regularly use hundreds of thousands to more than a million processors, this approach is not feasible. Instead, a local failure should permit a local recovery.

  One local-failure-local-recovery (LFLR) model permits the user to store specific data *persistently* for each MPI process and allows a recovery function to be registered, such that if a process fails, a new process is started and assigned to the rank of the failed process. The user's recovery function is then called, giving access to the persistent data of the old process as well as the neighbors' persistent data. Using LFLR, one can develop new algorithms for many types of problems.

- **Selective Reliability Programming**
  Selective reliability programming is another potential programming model in which the programmer has the ability to declare specific data and to compute some regions to be more reliable than the "bulk" reliability of the underlying system (alternatively, the default could be highly reliable with selectively less reliable regions). By distinguishing between what needs to be highly reliable or not, new algorithms can be developed that store most data and do most computations with low reliability while retaining the robustness of a fully reliable approach.

  Although the costs of high reliability will impact the practicality of some approaches, the details the implementation of reliability are not fundamentally important to reasoning about new algorithms. In some cases, even very expensive approaches, such as triple modular redundancy, can still be much faster than a fully unreliable approach.

Resilient computing models enable reasoning about and implementing a large collection of new algorithms, while also making existing ABFT approaches easier to implement. These kinds of models permit us to address the perceived resilience concerns of future computing systems. Preliminary work shows that expertise in applied mathematics, numerical algorithms, and floating-point arithmetic is essential to designing effective new algorithms.

4.6.2 Reproducibility

Today, many DOE applications use bitwise reproducibility as a surrogate for rigorous verification and validation, often at the behest of regulatory agencies. Bitwise reproducibility will be expensive if not impossible to achieve on exascale machines because it requires deterministic behavior, which is difficult to achieve in the presence of fault recovery and dynamic task scheduling. Requirements for bitwise reproducibility will need to be relaxed and will most likely need to be replaced with statistical concepts. Research into characterizations of expected variability in computed results will be necessary both to enable debugging at scale and to satisfy regulatory constraints. With regard to the former, one cannot overstate the usefulness of strict local and global conservation theorems, at least in some simplified geometries (e.g., periodic domains). The availability of such mathematical theorems in a discrete context is an invaluable tool to root out coding mistakes and thus to provide an important measure of correctness. Global sensitivity analysis, design of experiments, and other ingredients from uncertainty quantification (see Section 4.2.2) can also be expected to play a role in characterizing the expected variability in a given algorithm or computation.

4.6.3 Verification

Simulation codes must undergo code verification tests to provide confidence in the computed results, even before considering additional concerns such as validation and uncertainty quantification. Verification, as defined in [28], is "the process of determining, as completely as possible, whether a computer code correctly implements the intended algorithms, and determining the accuracy with which the algorithms solve the intended equations." While correctness might be seen as primarily a computer science concern, part of the appeal of bitwise reproducibility is that it provides a transference mechanism from one implementation whose correctness has been established to a new implementation, as in the case where a parallel implementation is required to exactly reproduce the results of a sequential implementation. In the absence of bitwise reproducibility, a new mechanism to efficiently establish the correctness of new implementations of floating-point computations must be developed. Analyses of solution accuracy should demonstrate not only that the code converges to the correct answer but also that the code converges at the expected rate; both are invaluable in code debugging.

Important for multiphysics and multiscale simulations are mechanisms for establishing the correctness of complex, integrated applications constructed from individual components whose correctness has been established. Code (order) verification is the preferred approach to demonstrating correctness, but this requires a known solution and therefore often fails to test the more complex interactions in the code. The method of manufactured solutions can generate more complex, integrated tests, but tools are needed to automate its use and codes must be developed with the necessary infrastructure to support this approach. This more systematic approach has not generally been attempted for non-mesh-based methods, so this represents a potential area for new research. Further, for multiscale applications using scale-bridging algorithms, one must perform verification across the model hierarchy, both as standalone models (to isolate errors in each level) and as a coupled system. The latter, however, may prove problematic unless one ensures that descriptions are discretely consistent across levels, namely, that different levels of the model hierarchy do not pollute the solution with separate truncation error contributions, for instance by enslaving numerical truncation errors across the hierarchy to reproduce that of a chosen level of the hierarchy. Early work in this area indicates that discrete consistency is in fact a major element of long-term nonlinear stability in scale-bridging algorithms and will likely prove essential in verification as well.

Even once a suitable set of test problems is established, the error still needs to be measured. Such measurement is often done by mesh convergence studies where all other code behavior is meant to be held fixed. The execution of such studies at scale is difficult now; but at the exascale will be even worse because of dynamic scheduling, dynamically changing multiphysics modules and models, and fault handling. Results will be difficult to interpret. Techniques to assess code convergence should be considered in order to understand dynamic code behavior and the effectiveness of test problems. Certain studies can be done at smaller scale and on individual components, but these will not necessarily characterize the behavior of the full code with the many complexities meant to optimize exascale performance, in particular, the effects of coupling errors.

New approaches to verification must be considered. Theoretically justifiable statistical approaches to convergence studies may be necessary. The exascale-motivated rewrite of applications is an opportunity to build a posteriori error estimation techniques into application codes, but these techniques need further research in order to be applicable to the anticipated multiscale, multiphysics models. The effects of possible faults in these error estimators must also be investigated.

4.6.4 Related Position Papers

Position papers related to resilience and correctness that were presented at the Exascale Mathematics Workshop include [WP70] and [WP56].

4.7 Mathematics for Exascale System Software

The impacts of mathematics research will not be confined to applications designed for exascale computing; the operation of the exascale machines will also benefit from new mathematically motivated approaches. As the system software stack becomes more adaptive and self-aware, the need for mathematical analysis and algorithms increases. No longer can ad hoc heuristics be expected to work reliably; methods with a solid theoretical foundation should be employed wherever possible. Mathematical and statistical techniques, particularly from optimization, can contribute to these dynamic system management challenges. Here we discuss four areas in which additional research is needed.

4.7.1 Autotuning Search as Derivative-Free Optimization

The search phase of autotuning can be posed as a derivative-free optimization problem and solved with existing optimization algorithms adapted to this new context or with wholly new algorithms developed to account for discrete design variables and hidden constraints. However, additional research is required in order to deal with the likely characteristics of exascale computing systems. In particular, new algorithms are needed to address multiple objectives such as (expected) execution time, memory footprint, resilience, and power demands. Furthermore, autotuning methods already struggle in situations that exhibit high variance in the stochastic response, often due to contention for shared resources. As power limits and other design and operational constraints lead to greater contention among and within jobs, we can expect this challenge to become more pronounced. Stochasticity will have to become a more explicit aspect of the autotuning process.

4.7.2 Adaptive Runtime Systems as Optimal Control Problems

Self-aware runtime systems monitor job performance and allocate more or fewer resources based on the job's performance goals and behavior, subject to constraints on available resources. To optimize performance, such systems rely on adaptive feedback control systems and machine learning. New research is required in order to address the case of additional, possibly interdependent, performance goals, such as power, time, and energy. In addition, scaling up to large computing systems demands decentralization and autonomy among the self-aware runtime agents. Understanding and controlling the expected behavior of such systems of agents will require game theoretic analysis and the development of distributed, multilevel optimization algorithms.

4.7.3 Mathematically Grounded Scheduling

Current batch scheduling systems frequently rely on fast heuristics to schedule normal jobs and perform backfilling. As scheduling becomes increasingly complicated, with a requirement to map jobs to heterogeneous resources subject to power and energy constraints, simple heuristics will likely not provide the level of system utilization demanded for leadership-class computing systems. Instead, it is likely that advanced job schedulers will rely on techniques from operations research to perform job scheduling and to estimate the actual resource requirements of jobs. New scheduling algorithms will, however, need to account for the highly adaptive and self-aware execution environments in which jobs are expected to execute, as well as the complex work flows typical of high-level analyses such as uncertainty quantification. Schedulers also need to account for the notoriously unreliable user estimates of execution time (even when incentives are provided for supplying an accurate expected value in addition to an upper bound). Consequently, effective scheduling algorithms can be expected to include statistical and machine learning models.

4.7.4 Stochastic Performance Models

Current performance models are typically deterministic or exhibit low variance when evaluated at configurations where experimental performance data has been collected. However, as algorithms and runtime systems become increasingly asynchronous and adaptive, we anticipate a greater demand for stochastic performance models. Such models can be used as a surrogate during autotuning, for extrapolation to predict performance and diagnose performance problems, and for use in anomaly detection. However, the construction of useful, mathematically justified performance models will require a close collaboration between computer scientists and applied mathematicians/statisticians.

4.7.5 Related Position Papers

Although important, the mathematics behind the dynamic management of exascale systems was not discussed explicitly at the Exascale Mathematics Workshop. One related position paper on fault detection that was not presented but is related is [WP4]. Several position papers from the Exascale OS/R Workshop are relevant to this discussion, including papers on optimization, optimal control, and machine learning [40, 52, 55], and a paper on machine learning from the performance modeling and simulation workshop [66].

5 Interdependencies with Other Efforts

Efforts to advance mathematics for exascale will not occur in a vacuum; other efforts sponsored by DOE ASCR and NNSA exist that either are directed toward the exascale computing goal or are relevant to some of the issues an exascale mathematics research program will address. In order to make the most effective use of the limited funding resources expected to be available, new research directions advocated in this report must be closely aligned and coordinated with other activities in the developing exascale ecosystem. Mathematicians will need to work with domain scientists and computer scientists to develop compatible, integrated approaches that leverage the latest development in hardware and software technologies while maintaining the overarching goal of enabling innovative science and engineering.

5.1 Existing DOE Efforts

Primarily led by DOE ASCR, there has been an increasing investment in the United States over the past five years in efforts that are both directly and indirectly relevant to exascale computing:

- **Exascale Co-Design Centers:** Co-design is a holistic design process where integrated teams of hardware architects, system software developers, domain scientists, computer scientists, and applied mathematicians work together to collaboratively develop compatible software and hardware solutions. It is an opportunity not only for the software and application side to reason about how to leverage emerging architectures and technology but also for the hardware developers and vendors to better understand the needs of DOE scientific computing. DOE ASCR has funded three exascale co-design centers, each organized around a specific DOE-relevant science area: combustion [41], materials [42], and nuclear reactors [18]. The applied mathematics community can learn from context provided by the co-design centers, and the co-design centers will benefit from advances in numerical algorithms developed by applied mathematics research efforts.

- **X-Stack:** Many issues need to be addressed for the exascale across the software stack. The ASCR X-Stack program supports nine projects [5] exploring innovative solutions to improve the programmability of exascale systems. Research areas include support for domain-specific languages, hierarchical programming models, compilers and compiler optimizations, adaptive runtime systems, execution models, autotuning frameworks, and support for resilience and fault containment. Applied mathematics efforts to develop new algorithms for the exascale will fundamentally rely on these technologies, and the X-Stack projects must understand the needs of numerical algorithms and application developers.

- **Exascale Operating and Runtime Systems:** The scale and complexity of exascale platforms are expected to require radically new functions and interfaces for system control, management of resources, communications, thread management, synchronization, power management, fault recovery, configuration, monitoring, and load balancing. These changes are being addressed in the Exascale Operating and Runtime Systems (OS/R) initiative launched in FY13 with the goal of developing a complete, platform-neutral prototype exascale OS/R. Two three-year projects were funded: Argo [4] and Hobbes [49].

- **FastForward and DesignForward:** The objective of the FastForward and DesignForward initiatives is to build partnerships between DOE and multiple computer hardware vendors in order to accelerate the research and development of critical technologies needed for extreme-scale computing. Research in these two-year projects involves all aspects of computer hardware, including innovative processor, memory, file system, and interconnect design, with goals to minimize energy use, maximize parallel performance, and ensure reliability. Applied mathematics research projects need to be aware of these advances in new technologies that may mitigate some of the anticipated exascale challenges. Similarly, the FastForward and DesignForward projects need to be informed about the needs of numerical algorithms and applications.

- **SciDAC Institutes:** The current SciDAC program is a five-year program that began in 2011 and is funded by DOE ASCR. SciDAC is not an exascale research program, but it is a major source of interaction in DOE computational science between the applied mathematics and science applications communities. Institutes are one component of SciDAC, and these have a mission to develop tools and resources that will enable and accelerate scientific discoveries through the use of advanced computing. The development is intended for the next five years of computer systems at the Oak Ridge and Argonne leadership computing facilities and at the National Energy Research Scientific Computing Center. While the SciDAC Institutes are focusing on computing platforms that are available near term, the research and development within these institutes may be relevant to the exascale machines being considered in this report; the current generation of petascale machines already introduce, to a lesser extent, some of the challenges of exascale, such as heterogeneous architectures, the availability of multi- and many-core systems, relatively small memory per core, resiliency, and power consumption.

  Two SciDAC Institutes are concerned primarily with applied mathematics research:

  - *FASTMath (Frameworks, Algorithms, and Scalable Technologies for Mathematics):* The FASTMath SciDAC Institute develops and deploys scalable mathematical algorithms and software tools for reliable simulation of complex physical phenomena and collaborates with DOE domain scientists to ensure the usefulness and applicability of FASTMath technologies.
  - *QUEST (Quantification of Uncertainty in Extreme-Scale Computations):* The QUEST SciDAC Institute focuses on uncertainty quantification in large-scale scientific computations. The overarching goal is to provide modeling, algorithmic, and general uncertainty quantification expertise, together with software tools, to other SciDAC Institutes, SciDAC applications, and Office of Science projects in general—thereby enabling and guiding a broad range of uncertainty quantification activities in their respective contexts.

- **ASCR Applied Mathematics Subprogram Initiatives and Projects:** In the past few years, many of the initiatives within ASCR's Applied Mathematics subprogram have required some consideration of future HPC architecture challenges or have considered the solution of increasingly complex multiscale, multiphysics problems, particularly in the context of larger design and decision questions.

  - *Mathematical Multifaceted Integrated Capability Centers:* To address grand challenges of increasing complexity within DOEs mission areas, ASCR established three Mathematical Multifaceted Integrated Capability Centers (MMICCs) to foster new integrated, iterative research processes across multiple mathematical disciplines. Started in FY12, these centers address holistically mathematics for scientific discovery, design, optimization, and risk assessment. The application targets for these centers are mesoscale modeling for materials, chemistry, and biofuels; complex energy systems such as the power grid; and multiscale, multiphysics modeling of subsurface flow and materials for energy storage. While exascale was not a primary focus of this call, sensitivity to next-generation HPC resources was considered. The greater relevance of this call is that it provides examples of integrated mathematics research activities across the "math stack," from problem formulation to analysis. Additional details can be found in the report [1].

– *Resilient Extreme-Scale Solvers:* To enable scientific discovery on the supercomputers expected to come online in the next 5–10 years and to lay the foundation for research in numerical algorithms for extreme-scale scientific computing, ASCR initiated the Resilient Extreme-Scale Solvers program in FY12 (although the projects did not officially start until late in FY13). The goal of this solicitation was to fund basic research that advances the state of the art in scalable, resilient, extreme-scale numerical algorithms. Four projects were funded, addressing topics in linear algebra solvers, nonlinear solvers, Monte Carlo algorithms, and high-order discretizations and related solvers (including hyperbolic and particle methods). Additional details can be found in the report [3].

– *Uncertainty Quantification Methodologies for Enabling Extreme-Scale Science:* ASCR's Uncertainty Quantification Methodologies for Enabling Extreme-Scale Science initiative, which was launched in FY13, focuses on basic research in methodologies and tools that will deliver advanced UQ capabilities for DOE-mission science while also anticipating the changes and challenges of using extreme-scale computing systems. Six projects funded in this solicitation address a variety of UQ topics, such as Bayesian inference, Markov-Chain Monte Carlo, multilevel methods for UQ of multiscale, and stochastic expansions, within the context of the anticipated challenges of the next-generation HPC architectures.

5.2 International Exascale Efforts

Internationally, the European Union, Japan, and China are also actively ramping up activities in exascale computing research. All three have independent funded efforts to build an exaflop machine by 2020, although Europe and Japan likely have the edge in developing the algorithms and software necessary to obtain exascale performance. In particular, the European Union's European Exascale Software Initiative (EESI) is a consortium of more than thirty academic institutions, government research laboratories, and private corporations organized to provide recommendations on strategic European actions with a particular focus on software improvement, cross-cutting issues advances, and gap analysis. The EESI is organized into eight working groups including Education and Strategic Coordination (including Co-Design), Applications, Enabling Technologies (including Numerical Libraries, Solvers, and Algorithms), and Cross-Cutting Issues (including Resilience). Two more specific goals of interest are to (i) produce a roadmap for transition of numerical libraries, the software eco-system, scientific software engineering, and programmability and (ii) promote an International Exascale Software Initiative within the international community. A clear connection exists between the EESI and the still-organizing U.S. exascale effort, and opportunities should be found to coordinate U.S. efforts with international activities where it makes sense to do so.

5.3 Areas of Collaboration with Other Exascale Efforts

Mathematics research for the exascale cannot proceed independent of other exascale activities. The applied mathematics community requires technical information, models, tools, and infrastructure from other research efforts within the DOE exascale ecosystem, and these efforts should not proceed without feedback and requirements from the applied mathematics community. Here we consider seven areas represented within the DOE exascale research activities and highlight the interdependencies of each area with applied mathematics research.

5.3.1 Architectures and Performance Modeling

To design numerical algorithms better suited for exascale simulation, the applied mathematics community needs information about those architectures or at least representations of them. No one has access to an exascale system yet. Therefore, new abstract machine models, such as those used in roofline analyses, are needed to guide thinking about the various trade-offs between algorithms and implementations; it is no longer adequate to consider crude estimates of operation complexity and memory usage. In order to demonstrate that performance gains are not merely theoretical, simulators that incorporate the latest exascale architectural concepts must be made available to numerical algorithm developers. In the area of resilience, any capabilities in hardware that can help detect soft faults will be beneficial in reducing fault tolerance to a problem of rapid recovery.

In the opposite direction, system architects and performance modelers need to better understand how exascale machines will be used. Applied mathematicians must provide clear descriptions of the nature of the discrete problems to be solved. Essential and negotiable characteristics of the problems and resulting algorithms must be identified. For instance, it is unlikely that stencil operations on multidimensional arrays can be completely abandoned. Thus, hardware support in the form of sophisticated prefetchers and advanced memory concepts that facilitate access to mutidimensional array sections can be expected to pay dividends. A dialogue is critical: today's data access patterns represent a particular design choice, and future choices will be dictated by a combination of algorithmic requirements and hardware constraints. Similarly, the coupling and communication of simulations often reflect fundamental physical constraints imposed on the problem. Where these couplings are artificial, they should be removed, but in many cases they must remain. Exascale architectures cannot assume that mathematical models will be found that completely eliminate global communication.

5.3.2 Operating and Runtime Systems

Operating system and runtime systems (OS/R) are the layers that insulate applications from the detailed complexity of the hardware through abstractions. It is extremely important that these abstractions provide interfaces that enable the more complex mathematical algorithms expected for good exascale performance. Applications will most likely depend on dynamic software composition, load balancing, and task scheduling throughout the execution of a simulation. Various levels of APIs for resource management (e.g., power, resilience, memory, fine-scale thread management) will need to be made available to numerical algorithms in order to allow for as little or as much user control as desired. Standardized APIs and interoperability of programming models are beneficial for mathematical algorithm and software development, but mathematicians must make their requirements known to the computer scientists developing these abstractions.

The input from applied mathematics to OS/R efforts, however, should go beyond needs and requirements for interfaces and capabilities. As explained in Section 4.7, applied mathematics can provide optimization techniques for autotuning of software and adaptivity of runtime systems and operations research approaches to dynamic scheduling. A very tight collaboration between computer scientists and applied mathematicians in OS/R research may prove very beneficial.

5.3.3 Programming Environment

A mathematical algorithm is only truly useful if it can be expressed in the form of efficient and maintainable software. The shape of the programming environment for the exascale is in flux, with numerous ideas about programming models, domain-specific languages, and compiler optimizations being proposed and investigated. Ideally, the developer of mathematical software would not need to deal with the complexities of memory layout, code fusion, transfer to and from accelerators, and so forth. Code could be developed in a maintainable, modular way by using portable parallel library interfaces (with machine-specific implementations hidden behind consistent interfaces) and intelligent compilers that would transform the code from its maintainable implementation into a formulation for efficient execution. Applied mathematicians must work with the designers and developers of the programming environments to help them understand the use patterns in numerical software, to help define suitable APIs, and to help define suitable compiler or precompiler commands that will allow the numericist to express optimization directives as easily and concisely as possible. Even if the ideal is not achieved in all cases, close collaboration between mathematicians and computer scientists can help ensure that mathematical software is as maintainable and performant as possible.

5.3.4 Development and Performance Tools

The complexity of exascale systems and application software makes it difficult to understand code (mis)behavior. Indeed, without an appropriate set of tools, once a simulation launches, it is difficult to understand the code execution and where the code might be failing or underperforming. Debugging at scale (perhaps with a million or more cores) of an application that is dynamically task-scheduled is even hard to conceptualize. Applied mathematicians will need a suite of exascale tools, not only for debugging, but also for understanding power usage, dynamic memory layout across deep memory hierarchies, data transfer patterns, and so on. Mathematicians must work with these tool developers to help them understand what types of diagnostics are necessary, which types may be specific to different classes of algorithms, and how the diagnostic information will be used, so that appropriate visualizations and data exploration tools can be devised.

5.3.5 Fault Management

Fault management touches on all aspects of exascale machine use. At many levels—the hardware, the OS, the runtime, the libraries, and the application itself—there are roles and responsibilities for fault management. Fault detection can occur at any level; by default, faults should be detected at the lowest possible level, ideally in hardware. However, when faults can be easily detected or tolerated at higher levels, it is desirable that low-level detection can be disabled in order to save time and energy. Once faults are identified, responsibility for dealing with the fault could occur anywhere throughout the stack. In some cases the application may choose to recover; in other cases the algorithms may be robust to the fault, making recovery unnecessary. Clearly, some kind of backplane for fault handling at various levels is necessary, and applied mathematicians must influence its design and development based on the tolerance or recovery characteristics of the algorithms and models used. In addition, applied mathematicians need software tools to aid in local restart and recovery; mathematics may provide techniques that allow local reconstruction or rollback without loss of accuracy, but the underlying software to change the execution path dynamically and locally and/or to execute recovery from some distributed checkpoint data must be provided by the computer science community.

5.3.6 Data Management

Of course, the point of simulation for DOE-relevant problems is to generate useful data to help understand or solve important science and engineering problems. Schemes must be developed for handling this data efficiently and without losing important artifacts. Data management includes workflow systems; metadata generation and capture; data representations; data movement on and between machines; and the ability to manipulate, annotate, archive, and share data. At the exascale, these activities will be challenging. More data, from both higher resolutions and ensemble simulations, and the increasing disparities between computational capacity and I/O infrastructure performance will necessitate new data-processing modes, in particular, in situ data staging and processing, that complement traditional data postprocessing. a

From the applied mathematics perspective, through analysis and visualization, applied mathematics is already well entwined with data management. In situ mathematical analysis and data reduction techniques will help address some of the data management challenges. These same algorithms will rely on the APIs and infrastructure developed for exascale data management.

5.3.7 Applications and Co-Design

Applied mathematics provides the mapping from a physical model to computer hardware. This mapping is accomplished through a set of choices at each level of the *mathematics stack*. Applied mathematicians must select the mathematical model to use, including UQ and optimization formulations; the discrete representation; suitable solvers and algorithms; and the data analysis to be performed. At a basic level, many of these choices must be informed both by the application requirements and by the underlying hardware. A holistic co-design cycle must involve applied mathematicians and consider the interplay among applications, algorithms, and architectures. There must be a close collaboration between the applied mathematics community and application co-design activities so that the former better understand the needs of domain scientists and so that the latter are aware of new advances in exascale mathematics.

6 Common Themes, Findings, and Recommendations

Historically, advances in computational mathematics have contributed as much to increases in high-performance computing as have improvements in hardware. In the move to exascale computing, this situation will not change. Mathematics is intimately involved in numerical simulation and in design and decision problems, from the problem formulation and modeling, through discrete algorithms and data analysis, to system operations. From our consideration of the role of mathematics in exascale computing research, we present here the common themes, high-level conclusions, and recommendations for a path forward for the DOE ASCR program.

6.1 Themes

Three common themes emerged from the information that the working group collected: hierarchies, integrated and holistic approaches, and adaptivity and automation. We expect these three themes to permeate exascale mathematics research, and we review them here as useful paradigms to guide thinking about approaches for simulation on exascale computers.

Hierarchies. The idea of hierarchies in exascale computing is pervasive. The architecture of exascale machines will be hierarchical: multiple nodes connected by an interconnect, where each node comprises multiple processing units and/or accelerators with multiple types and levels of memory. Similarly, multilevel or hierarchical models and algorithms are expected to play a significant role in the formulation and efficient solution of science problems on exascale architectures. Multiscale models can and should be formulated in hierarchical ways; this structure can be harnessed to allow for asynchrony, to communicate information efficiently across the solution, to accelerate the fine-scale solution, and to map different models to the most congruent architectural layer (e.g., moment models on CPUs and particle models on GPUs). Multilevel methods for linear algebra are mathematically optimal; for many DOE applications, these methods form the basis for the only truly scalable solvers. Hierarchical representations of solutions also provide natural opportunities for algorithmic-based fault tolerance. Collective operations, which are common in scientific applications, may be most efficiently implemented through tree-based hierarchical algorithms.

Integrated and holistic approaches. The challenges of exascale computing cut across traditional discipline domains. For this reason, DOE has already established three exascale co-design centers, which are meant to bring together application scientists, applied mathematicians, and computer scientists and engineers to solve the exascale design problem collaboratively while respecting the requirements and constraints of the problems, the algorithms, and the hardware. In a similar way, the mathematical areas represented in the *math stack*, from problem formulation through data analysis, should be considered holistically because there are interdependencies throughout the stack. Choices made in formulations, models, and discretizations, for instance, constrain the possible solution methods and parallel implementations, while the availability of scalable solver algorithms places limits on problem formulation and model choices.

Adaptivity and automation. Exascale machines will be architecturally more complex, as will the operating systems and runtime environments, which will need to ensure reliability and promote efficient machine usage (power, throughput, etc.). Given the vast number of components (e.g., millions to billions of cores and deep memory hierarchies), this complexity cannot be managed manually by human operators. The systems and their software will therefore require a great deal of adaptivity and automation.

Similarly, adaptivity and automation will play key roles in scientific simulations and the algorithms that enable them. Adaptive mesh and model refinement will reduce problem size and concentrate resources where better fidelity is required. Comparable adaptivity will likely occur in optimization and UQ algorithms in order to reduce the search time and/or size of the parameter space. Resilience will likely require simulations to migrate or restart locally and on the fly, instead of the traditional fail-stop model. All these examples present dynamically changing, heterogeneous workloads that will require automated, dynamic load balancing to maximize parallel efficiency.

Automation will also play an important role in numerical software portability. Portability will increasingly be a challenge as different hardware configurations are tried on the path to achieve exaflop-capable machines. Maintaining software optimized for any given architecture, and optimizing for each new architecture, will be cost-prohibitive. Libraries of numerical algorithms can address this challenge through autotuning.

6.2 Findings

Based on our inquiry, the Exascale Mathematics Working Group reports the following six findings.

Finding 1: *Exascale computing will enable us to use computation to solve problems in ways that are not feasible today and will result in significant scientific breakthroughs. However, the transition to exascale poses numerous scientific and technological challenges.*

Reaching exaflop performance on a limited power budget will come at the cost of a dramatically altered computer architecture that will require substantial reconsideration of the algorithms involved in simulation. However, one must avoid the trap of focusing solely on the architectural challenges associated with exascale computing and of forgetting the prime reason to pursue it: the new science that will be enabled. DOE has a mission to solve some of the most challenging scientific problems our nation faces, and algorithmic research is needed in order to address these extreme-scale problems. Examples include designing novel materials and chemical processes at the nanoscale to produce specified macroscale behavior, incorporating information about fine-scale environmental processes into models of climate, and designing next-generation energy generation and conversion technologies to meet growing energy demands.

Finding 2: *Without a close collaboration between applied mathematicians, computer scientists, and application scientists, we will not be able to develop a computational science discovery environment capable of exploiting the computational resources that will be available at the exascale.*

In simulation, the choice of problem formulation and mathematical model is in part motivated by the science objectives, but it is also constrained by the computer hardware and the set of known available algorithms. Understanding the breadth of the requirements and constraints and finding solutions that balance these will require multidisciplinary teamwork. Domain scientists must work with mathematicians to formulate problems, models, and discretizations that are tractable for discrete solvers. Mathematicians must work with computer scientists and engineers to develop new algorithms and implementations that can efficiently harness architectural features. Computer scientists must collaborate with domain scientists and mathematicians to ensure that programming environments, runtime environments, and performance measurement tools provide functionality relevant to their needs. Exascale computing is forcing an end-to-end reconsideration of high-performance computing, and close collaboration will be necessary to converge more rapidly on a useful simulation environment.

Finding 3: *Advances in applied mathematics, in areas such as mathematical modeling, numerical analysis, and adaptive algorithms, will be essential in order to produce high-performance exascale applications and will provide key input to application scientists and computer scientists.*

A great task before us is to determine those models and algorithms that will be successful for use in exascale computing, including existing models and algorithms and new ones yet to be devised. The performance of current algorithms on hypothetical exascale systems must be understood in order to find improved implementations and to motivate new algorithms. Opportunities must be provided to explore new problem formulations and discretization approaches. New discretizations that provide higher order, tighter coupling, and the discrete preservation of invariant and asymptotic properties will be needed. Adaptivity in models, mesh, sampling, and configuration will be needed in order to make best use of the exascale computers, and new algorithms must be devised that make use of adaptive procedures. Furthermore, it is not enough to devise schemes that produce answers quickly; indeed, it is of no use if these answers are wrong. Numerical analysis, which provides proof (or at least justification) of the consistency, accuracy, and stability of numerical algorithms, must be advanced to address the more advanced algorithms expected for exascale computers. Simulation, at its core, is applied mathematics; and the results of this applied mathematics research will inform and modify the thinking of domain scientists and computer scientists who will use advances in computational mathematics to develop exascale-capable applications and to motivate new capabilities in the development and runtime environments.

Finding 4: *Exascale computing will enable a more holistic treatment of complex problems.*

Exascale provides an opportunity to move beyond the loosely coupled, forward-simulation paradigm that has driven much of scientific computing up through petascale. This change contains the "revolution" in mathematics needed for exascale computing: not a "new mathematics," but a *rethinking* of the way we design algorithms and codes for better physical fidelity and to address new questions. Because of the architectural challenges, we anticipate a need to invest in a new generation of codes, and this opportunity should be seized to incorporate better algorithms and support for UQ and optimization. For application domains that are prepared for these advanced formulations, exascale platforms offer the first concrete opportunity to improve the value of simulation qualitatively.

Finding 5: *Because computer architectures will be altered from supercomputers down through personal computers, some advances in algorithms devised for the exascale likely will benefit computation across the range of resources.*

Computer architectures are changing across the board with the capability to add more cores to computer chips and a desire for low-power computing from cell phones up to the exascale. Design improvements at both ends of the hardware spectrum can have far-reaching impacts. Accordingly, some algorithms intended to address the challenges of exascale computing will be relevant to machines at smaller scales of computing, and fundamental algorithmic research at small scale could also have unexpected impact on exascale simulation.

Finding 6: *In addition to fundamental research into new algorithms, resources will be needed for more applied research and development to extend the large collection of existing DOE mathematics libraries so that they make better use of exascale architectural features.*

Libraries are powerful means of sharing verified, optimized algorithms; accordingly, DOE has invested in the development of numerous numerical libraries. Such investments must continue, and these investments must go beyond funding the development of new algorithms. Substantial software engineering costs must be borne in the extension of existing libraries to the exascale. Furthermore, autotuning of libraries will play a more significant role at the exascale, since libraries will need to support a wider variety of platforms and automated empirical discovery and optimization will accelerate scientific computing workflows. Developing autotuning capabilities will require additional up-front investment, with the promise of long-term savings in user and computer time.

6.3 Recommendations

Our findings indicate that the DOE Advanced Scientific Computing Research program needs to take action to build a more explicit research program in applied mathematics for exascale computing. We summarize the necessary actions in five key recommendations.

Recommendation 1: *DOE ASCR should proceed expeditiously and with high priority with an exascale mathematics initiative so that DOE continues to lead in using extreme-scale computing to meet important national needs.*

This report demonstrates a clear need for research in applied mathematics specifically for exascale computing. The promise of exascale computing will not be realized without advances in computational mathematics. Research in applied mathematics often takes years to produce results, and the implementation of those results requires additional time. The R&D that leads to those advances can and must start now. We cannot wait until the first exaflop machine exists; only with preparation and investment will we be able to make efficient use of such a machine soon after its arrival.

Recommendation 2: *A significant new investment in research and development of new models, discretizations, and algorithms implemented in new science application codes is required in order* *to fully leverage the significant advances in computational capability that will be available at the exascale.*

The move to exascale will be a more disruptive transition than the previous moves to terascale and petascale because of the significant changes in computer architecture. Many existing algorithms and implementations that have relied on steady clock speed improvements cannot exploit the performance trends of future systems. Current algorithms will need to be evaluated and either modified or abandoned; alternative algorithms will need to be invented or rediscovered. The promise of exascale resources will also provide opportunities for other models and problem formulations, but how and when to use such models are open questions. Much investigation needs to be done on specific classes of models and algorithms. Because of the complexity of the problem and the numerous trade-offs, it is too early to designate any one technology or approach as superior; therefore, a diverse research portfolio is the most robust investment strategy at this point. In order to establish such a diverse program, a broad segment of the applied mathematics community must be engaged, and hence funding opportunities must be created that enable broad participation.

Recommendation 3: *Not all problems require exascale computation, and yet these problems will continue to require applied mathematics research. Thus, a balance is needed in the DOE applied mathematics research portfolio that provides sufficient resources to realize the potential of exascale simulation while preserving a healthy base research program.*

Most science and engineering are not done at the most extreme scale of computing, yet nevertheless require sophisticated algorithms, improved time to solution, and better model fidelity. We expect that because of some similarities in future architectures, new approaches for the exascale will improve smaller-scale applications. However, this trickle-down benefit cannot be the only support provided to scientists not working at the exascale. Resources must continue to be invested in applied mathematics areas that are not focused on exascale computing. In addition to supporting the whole mission of DOE, fundamental research at small scale may result in models and algorithms that have unexpected impacts on exascale computing. Thus, new funding must be obtained to support an exascale mathematics research program, while preserving the current applied mathematics base program. The so-called Brown Report [15] identified priority areas of research in applied mathematics to support the DOE.

Recommendation 4: *An intensive co-design effort is essential for success, where computer scientists, applied mathematicians, and application scientists work closely together to produce a computational science discovery environment able to exploit the computational resources that will be available at the exascale.*

While research is needed into individual models and algorithms at the exascale, aspects of the problem suggest that a holistic approach should also be pursued. Many of the opportunities in exascale computing will come from reconsidering the entire problem—from formulation through analysis and co-designing models, discretizations, and solvers that work together. Of course, more fundamental research into each of the components is necessary so that options and trade-offs are well understood. Nevertheless, no single domain can work in isolation; in particular, mathematicians must understand the needs of the domain scientists and the computational environments in which simulations will be run. Like the challenge of putting a man on the moon, assembling a diverse team to focus on a well-defined goal, such as a scientific grand-challenge problem, has advantages over attempting to solve more general and therefore vaguely defined problems. This is true not just between the domain science, mathematics, and computer science disciplines but also within the applied mathematics specialties (i.e., across the math stack). Co-design and MMICCs-like projects should be pursued. However, in order to promote diversity in approaches and to engage as much of the applied mathematics community as possible, the DOE ASCR research portfolio cannot be limited to end-to-end projects; the limited number of such projects will limit the number of participating researchers.

Recommendation 5: *DOE ASCR must make investments to increase the pool of computational scientists and mathematicians trained in both applied mathematics and high-performance computing.*

In order to advance mathematics for exascale computing, a well-trained workforce is critical. Substantial research in applied mathematics is necessary for exascale computing, and so new research funds are required. There will be a corresponding need to grow the community capable of executing this new research program. Such researchers will require a breadth of understanding beyond applied mathematics. Exascale computers will introduce changes in system operation and program execution that, at least for some time, will not be hidden behind programming abstractions. Computational mathematicians will therefore need a better understanding of the computer science issues involved. Furthermore, the science needs that drive exascale computing, combined with the HPC system changes, present opportunities to rethink how and what we simulate. Progress in exascale simulation will be driven by integrated, holistic approaches that will require knowledge throughout the mathematics stack and beyond, into both the applications and the computer science. Efforts such as co-design will require multidisciplinary teams but will operate most effectively when those teams comprise researchers with interdisciplinary skills and knowledge. Thus, investments will be needed to develop the workforce necessary to execute an applied mathematics research agenda for exascale computing. This workforce development should provide opportunities both for new researchers (e.g., through interdisciplinary graduate fellowships) and for existing researchers in DOE and the greater applied mathematics community.

6.4 Impact

Applied mathematics research is a critical component of the overall exascale computing enterprise. Enhancing the national capabilities in advanced mathematical modeling, numerical algorithms, and software will have a major impact on our future national research capacity and an international impact in the ever-increasing number of domains within which high-performance computing is, or is set to become, a core activity. It is essential that DOE make strategic investments now in high performance-computing algorithms and software in order to enable successful use of exascale resources in support of its mission and to safeguard our ability to continue to lead the world in this field.

References

[1] F. Alexander, M. Anitescu, J. Bell, D. Brown, M. Ferris, M. Luskin, S. Mehrotra, B. Moser, A. Pinar, A. Tartakovsky, K. Willcox, S. Wright, and V. Zavala. Report of the DOE workshop on mathematics for the analysis, simulation, and optimization of complex systems: A multifaceted mathematical approach for complex systems. Report, U.S. Department of Energy, ASCR, March 2012. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Multifaceted_Mathematical_Approach_for_Complex_Systems.pdf.

[2] J. D. Anderson et al. *Computational Fluid Dynamics*, volume 206. McGraw-Hill New York, 1995.

[3] J. Ang, K. Evans, A. Geist, M. Heroux, P. Hovland, O. Marques, L. Curfman McInnes, E. Ng, and S. M. Wild. Report on the workshop on extreme-scale solvers: Transition to future architectures. Report, U.S. Department of Energy, ASCR, April 2012. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/reportExtremeScaleSolvers2012.pdf.

[4] Argo: An exascale operating system. URL http://www.mcs.anl.gov/project/argo-exascale-operating-system.

[5] ASCR X-Stack portfolio. URL http://science.energy.gov/ascr/research/computer-science/ascr-x-stack-portfolio.

[6] I. M. Babuška, R. Tempone, and G. Zouraris. Galerkin finite element approximations of stochastic elliptic partial differential equations. *SIAM J. Numer. Anal.*, 42(2):800–825, 2004.

[7] N. S. Bakhvalov. On the convergence of a relaxation method with natural constraints on an elliptic operator. *USSR Comput. Math. Math. Phys.*, 6:101–135, 1966.

[8] M. Balasubramanian, E. L. Shwartz, J. B. Tenenbaum, V. de Silva, and J. C. Langford. The Isomap algorithm and topological stability. *Science*, 2002.

[9] T. Belytschko, Y. Guo, W. K. Liu, and S. P. Xiao. A unified stability analysis of meshless particle methods. *Int. J. Numer. Meth. Engng.*, 48:1359–1400, 2000.

[10] J. Bennett, V. Krishnamoorthy, S. Liu, R. Grout, E. R. Hawkes, J. H. Chen, J. Shepherd, V. Pascucci, and P.-T. Bremer. Feature-based statistical analysis of combustion simulation data. *IEEE Trans. Vis. Comp. Graph.*, 17(12):1822–1831, 2011.

[11] M. J. Berger and J. Oliger. Adaptive mesh refinemenr for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, 1984.

[12] E. L. Bouzarth and M. L. Minion. A multirate time integrator for regularized Stokeslets. *J. Comput. Phys.*, 229(11):4208–4224, 2010.

[13] A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Math. Comp.*, 31: 333–390, 1977.

[14] A. Brandt, S. F. McCormick, and J. W. Ruge. Algebraic multigrid (AMG) for sparse matrix equations. In D. J. Evans, editor, *Sparsity and Its Applications*. Cambridge University Press, Cambridge, 1984.

[15] D. L. Brown, J. Bell, D. Estep, W. Gropp, B. Hendrickson, S. Keller-McNulty, D. Keyes, J. T. Oden, L. Petzold, and M. Wright. Applied mathematics at the U.S. Department of Energy: Past, present and a view to the future. A Report by an Independent Panel from the Applied Mathematics Research Community, May 2008. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/brown_report_may_08.pdf.

[16] J. C. Butcher. On the convergence of numerical solutions of ordinary differential equations. *Math. Comp.*, 20:1–10, 1966.

[17] J. C. Butcher. General linear methods. *Acta Numerica*, 15:157–256, 2006.

[18] CESAR: Center for Exascale Simulation of Advanced Reactors. URL http://cesar.mcs.anl.gov.

[19] Committee on Mathematical Foundations of Verification, Validation, and Uncertainty Quantification; Board on Mathematical Sciences and Their Applications, Division on Engineering and Physical Sciences, National Research Council. *Assessing the Reliability of Complex Models: Mathematical and Statistical Foundations of Verification, Validation, and Uncertainty Quantification.* The National Academies Press, 2012. URL http://www.nap.edu/openbook.php?record_id=13395.

[20] G. A. Dilts. Moving-least-squares-particle hydrodynamics–I. Consistency and stability. *Int. J. Numer. Meth. Engng.*, 44:1115–1155, 1999.

[21] DOE ASCAC Subcommittee Report. ASCAC Subcommittee Report: The Opportunities and Challenges of Exascale Computing, Fall 2010. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Exascale_subcommittee_report.pdf.

[22] DOE Workshop Report. Scientific Grand Challenges: Challenges for Understanding the Quantum Universe and the Role of Computing at the Extreme Scale, December 2008. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Hep_report.pdf.

[23] DOE Workshop Report. Scientific Grand Challenges: Challenges in Climate Change Science and the Role of Computing at the Extreme Scale, November 2008. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Climate_report.pdf.

[24] DOE Workshop Report. Scientific Grand Challenges: Discovery in Basic Energy Sciences: The Role of Computing at the Extreme Scale, August 2009. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Bes_exascale_report.pdf.

[25] DOE Workshop Report. Scientific Grand Challenges: Architectures and Technology for Extreme Scale Computing, December 2009. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Arch_tech_grand_challenges_report.pdf.

[26] DOE Workshop Report. Scientific Grand Challenges: Opportunities in Biology at the Extreme Scale of Computing, August 2009. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Biology_report.pdf.

[27] DOE Workshop Report. Scientific Grand Challenges: Fusion Energy Sciences and the Role of Computing at the Extreme Scale, March 2009. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Fusion_report.pdf.

[28] DOE Workshop Report. Scientific Grand Challenges for National Security: The Role of Computing at the Extreme Scale, October 2009. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Nnsa_grand_challenges_report.pdf.

[29] DOE Workshop Report. Science Based Nuclear Energy Systems Enabled by Advanced Modeling and Simulation at the Extreme Scale, May 2009. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Sc_nework_shop__report.pdf.

[30] DOE Workshop Report. Scientific Grand Challenges: Forefront Questions in Nuclear Science and the Role of Computing at the Extreme Scale, January 2009. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Np_report.pdf.

[31] DOE Workshop Report. Scientific Grand Challenges: Crosscutting Technologies for Computing at the Exascale, February 2010. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Crosscutting_grand_challenges.pdf.

[32] DOE Workshop Report. Exascale and Beyond: Configuring, Reasoning, Scaling, August 2011. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/ArchitecturesIIWorkshopReport.pdf.

[33] DOE Workshop Report. Scientific Discovery at the Exascale: Report from the DOE ASCR 2011 Workshop on Exascale Data Management, Analysis, and Visualization, February 2011. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Exascale-ASCR-Analysis.pdf.

[34] DOE Workshop Report. Exascale Programming Challenges, July 2011. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/ProgrammingChallengesWorkshopReport.pdf.

[35] DOE Workshop Report. Tools for Exascale Computing: Challenges and Strategies, Oct 2011. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Exascale_Tools_Workshop_Report.pdf.

[36] DOE Workshop Report. Fault Management Workshop, August 2012. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/FaultManagement-wrkshpRpt-v4-final.pdf.

[37] DOE Workshop Report. Exascale Operating Systems and Runtime Software Report, December 2012. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/ExaOSR-Report-Final.pdf.

[38] DOE Workshop Report. Report on the ASCR Workshop on Modeling and Simulation of Exascale Systems and Applications, August 2012. URL http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/ModSim_Report-2012_AH_5.pdf.

[39] A. Dutt, L. Greengard, and V. Rokhlin. Spectral deferred correction methods for ordinary differential equations. *BIT Num. Math.*, 40(2):241–266, 2000.

[40] C. Engelmann, G. Vallée, and T. Naughton. Dynamic self-aware runtime software for exascale systems. In *DOE Workshop on Exascale Operating Systems and Runtimes*, Oct. 2012. URL https://collab.mcs.anl.gov/display/exaosr/Position+Papers.

[41] ExaCT: Center for Exascale Simulation of Combustion in Turbulence. URL http://exactcodesign.org.

[42] ExMatEx: Exascale Co-Design Center for Materials in Extreme Environments. URL http://codesign.lanl.gov/projects/exmatex.

[43] R. P. Fedorenko. A relaxation method for solving elliptic difference equations. *USSR Comput. Math. Math. Phys.*, 1:1092–1096, 1962.

[44] G. Fishman. *Monte Carlo*. Springer-Verlag, New York, 1996. ISBN 0-387-94527-X.

[45] A. L. Garcia, J. B. Bell, W. Y. Crutchfield, and B. J. Alder. Adaptive mesh and algorithm refinement using direct simulation Monte Carlo. *J. Comput. Phys.*, 154(1):134–155, 1999.

[46] R. G. Ghanem and P. D. Spanos. *Stochastic finite elements: A spectral approach*. Springer-Verlag, New York, 1991. ISBN 0-387-97456-3.

[47] A. Griewank and A. Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM, Philadelphia, PA, 2nd edition, 2008. ISBN 978-0-898716-59-7.

[48] M. Gunzburger, C. G. Webster, and G. Zhang. Stochastic finite element methods for partial differential equations with random input data. *Acta Numerica*, 23, 2014.

[49] Hobbes: An operating system for extreme-scale systems. URL http://xstack.sandia.gov/hobbes.

[50] M. Hochbruck and A. Ostermann. Exponential integrators. *Acta Numerica*, 19:209–286, 2010.

[51] M. Hochbruck, C. Lubich, and H. Selhofer. Exponential integrators for large systems of differential equations. *SIAM J. Sci. Comput.*, 19(5):1552–1574, 1998.

[52] H. Hoffmann. Managing competing goals with a self-aware runtime system. In *DOE Workshop on Exascale Operating Systems and Runtimes*, Oct. 2012. URL https://collab.mcs.anl.gov/display/exaosr/Position+Papers.

[53] K.-H. Huang and J. A. Abraham. Algorithm-based fault tolerance for matrix operations. *IEEE Transactions on Computers*, C-33(6), 1984.

[54] S. Jin. Asymptotic preserving (AP) schemes for multiscale kinetic and hyperbolic equations: a review. *Rivista di Matematica della Universita di Parma*, 3:177–216, 2012.

[55] L. Kale. An introspective and adaptive runtime system. In *DOE Workshop on Exascale Operating Systems and Runtimes*, Oct. 2012. URL https://collab.mcs.anl.gov/display/exaosr/Position+Papers.

[56] D. E. Keyes, L. C. McInnes, C. S. Woodward, W. Gropp, E. Myra, M. Pernice, J. Bell, J. Brown, A. Clo, J. Connors, E. Constantinescu, D. Estep, K. Evans, C. Farhat, A. Hakim, G. Hammond, G. Hansen, J. Hill, T. Isaac, X. Jiao, K. Jordan, D. Kaushik, E. Kaxiras, A. Koniges, K. Lee, A. Lott, Q. Lu, J. Magerlein, R. Maxwell, M. McCourt, M. Mehl, R. Pawlowski, A. P. Randles, D. Reynolds, B. Rivière, U. Rüde, T. Scheibe, J. Shadid, B. Sheehan, M. Shephard, A. Siegel, B. Smith, X. Tang, C. Wilson, and B. Wohlmuth. Multiphysics simulations: Challenges and opportunities. *Int. J. High Perf. Comput. App.*, 27:4–83, 2013.

[57] P. Kogge, K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, K. Hill, J. Hiller, S. Karp, S. Keckler, D. Klein, R. Lucas, M. Richards, A. Scarpelli, S. Scott, A. Snavely, T. Sterling, R. S. Williams, and K. Yelick. Exascale computing study: Technology challenges in achieving exascale systems, 2008. URL http://www.cse.nd.edu/Reports/2008/TR-2008-13.pdf.

[58] J. L. Lions, Y. Maday, and G. Turinici. Résolution d'EDP par un schéma en temps pararéel. *C. R. Acad Sci. Paris Sér. I Math*, 332:661–668, 2001.

[59] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard, Version 3.0.*

[60] M. L. Minion. Semi-implicit spectral deferred correction methods for ordinary differential equations. *Comm. Math. Sci.*, 1(3):471–500, 2003.

[61] J. Nievergelt. Parallel methods for integrating ordinary differential equations. *Comm. ACM*, 7:731–733, 1964.

[62] F. Nobile, R. Tempone, and C. G. Webster. An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM J. Numer. Anal.*, 46 (5):2411–2442, 2008.

[63] W. L. Oberkampf, M. Pilch, and T. G. Trucano. Predictive capability maturity model for computational modeling and simulation. Technical Report SAND2007-5948, Sandia National Laboratories, October 2007.

[64] J. W. Ruge and K. Stüben. Algebraic multigrid (AMG). In S. F. McCormick, editor, *Multigrid Methods*, volume 3 of *Frontiers in Applied Mathematics*, pages 73–130. SIAM, Philadelphia, PA, 1987.

[65] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4:409–423, 1989.

[66] S. L. Song, N. R. Tallent, and V. A. Vishnu. Exploring machine learning techniques for dynamic modeling on future exascale systems. In *DOE annual workshop on modeling and simulation of exascale systems and applications (MODSIM)*, Sept. 2013. URL https://sites.google.com/site/modsimws2013/home/workshop-papers.

[67] H. Victory, Jr and E. J. Allen. The convergence theory of particle-in-cell methods for multidimensional Vlasov-Poisson systems. *SIAM J. Numer. Anal.*, 28(5):1207–1241, 1991.

[68] D. Xiu and J. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM J. Sci. Comput.*, 27:1118–1139, 2005.

[69] R. Zwanzig. *Nonequilibrium statistical mechanics.* Oxford University Press, 2001.

Submitted Position Papers

Electronic versions of these position papers are available for download from the Exascale Mathematics Workshop website: https://collab.mcs.anl.gov/display/examath/Submitted+Papers.

[WP1] Beth Wingate, “Computing Highly Oscillatory Systems at the Exascale.”

[WP2] Chi-Wang Shu, “Position paper: Discontinuous Galerkin methods for exascale computing.”

[WP3] Shengxin Zhu, “Small dots, big challenging?”

[WP4] Nageswara Rao, “Fault Detection and Profiling Algorithms for Exascale Computing Systems.”

[WP5] Wei Cai, “A new extreme-scale parallel Poisson/Helmholtz solver combining local boundary integral equation and random walk methods.”

[WP6] James Glimm, “Finite Rate Chemistry.”

[WP7] Nageswara Rao, “Confidence Estimation for Exascale Computations.”

[WP8] Jean-Luc Fattebert, “Short-range O(N) algorithms for First-Principles Molecular Dynamics at extreme scales.”

[WP9] Ulrike Yang, “Multilevel Methods Combine All Properties that are Essential for Fast and Efficient Performance at the Extreme Scale.”

[WP10] Wei Cai and Shanhui Fan, “Parallel stochastic methods for exploring randomness in solar cell designs.”

[WP11] Peter Brune, Barry Smith, and Matthew Knepley, “Nonlinear Solver Algorithms at the Exascale: Rethinking the Full Linearization Bottlenecks.”

[WP12] Kiran Bhaganagar, “Numerical simulation of turbulent flows in natural systems: Framework.”

[WP13] Eric Phipps, H. Carter Edwards, Jonathan Hu, and Clayton Webster, “Realizing Exascale Performance for Uncertainty Quantification.”

[WP14] Jeff Banks and Bill Henshaw, “The design and development of modular and adaptive algorithms.”

[WP15] Sven Leyffer, Todd Munson, Stefan Wild, Bart van Bloemen Waanders, and Denis Ridzal, “Mixed-Integer PDE-Constrained Optimization.”

[WP16] William Henshaw and Jeffrey Banks, “Applied Math Exascale Workshop Position Paper: A Challenge Problem Competition and Multi-Algorithm Optimizations.”

[WP17] Tzanio Kolev, “Scalable multi-physics simulations will require new discretization and numerical methods research.”

[WP18] Zhaojun Bai and Ren-Cang Li, “Scalable Eigensolvers for Excited States Calculations.”

[WP19] Lori Diachin, Mark Shephard, Jeff Banks, Tim Tautges, Onkar Sahni, and William Henshaw, “Scalable Mesh-Based Simulation Workflows are required for Exascale Computers.”

[WP20] Howard Elman and Bedrich Sousedik, "Toward Exascale Solvers for Stochastic FEM."

[WP21] Jinchao Xu, "Design and Analysis of Fault-tolerant Multilevel Iterative Algorithms."

[WP22] Wim Vanroose, Pieter Ghysels, Dirk Roose, and Karl Meerbergen, "Hiding global communication latency and increasing the arithmetic intensity in extreme-scale Krylov solvers."

[WP23] Valery Il'In and Dmitry Butyugin, "Exascale challenges and issues of applied mathematics."

[WP24] Yu Zhuang, Michele Ceotto, William Hase, and Wibe De Jong, "Towards Higher Scalability than Embarrassing Parallelism: Some Thoughts on Ab Initio Semiclassical Molecular Dynamics Simulations on Exascale Machines."

[WP25] Dana Knoll, Luis Chacon, Tim Kelley, and Kord Smith, "Moment-based scale-bridging algorithms for multiphysics kinetic simulations at the exascale."

[WP26] Yan Wang, "Reliable Extreme-Scale Stochastic Dynamics Simulation based on Generalized Interval Probability I. Uncertainty Dynamics."

[WP27] Assad Oberai, Onkar Sahni, and Mark Shephard, "Adaptive Multiscale Predictions at Exascale."

[WP28] Duan Zhang and Dana Knoll, "Tree-like Processor Architecture for Multi-scale, Multi-material and Multi-physics Computation Based on the Material Point Method."

[WP29] Charles Tong, "A Position Paper on Extreme-scale Uncertainty Quantification Methodologies."

[WP30] Yan Wang, "Reliable Extreme-Scale Stochastic Dynamics Simulation based on Generalized Interval Probability II. Multiscale Quantification."

[WP31] Michael Mascagni and Michela Taufer, "Monte Carlo Methods Are Perfectly Situated to Enable Exascale Scientific Computing."

[WP32] Bert Debusschere and Habib Najm, "Probabilistic Approaches for Communication Avoidance and Resilience in Exascale Simulations."

[WP33] Joseph Teran and Alice Koniges, "Material Point Methods and Multiphysics for Fracture and Multi-phase Problems."

[WP34] Eduardo D'Azevedo and Judith Hill, "External Memory Algorithms for Reducing Data Movement."

[WP35] Judith Hill and Bronson Messer, "Engagement of the Leadership Computing Facilities in the Exascale Math Conversation: A Critical Gap."

[WP36] Alice Koniges, Jean-Luc Vay, Alex Friedman, Hartmut Kaiser, and Thomas Sterling, "Consideration of Asynchronous Algorithms for Particle-Grid Simulations."

[WP37] Qiqi Wang, "Scalable Parallel-in-Time Simulation of Extreme-Scale Chaotic Systems."

[WP39] Brian Van Straalen, "Heterogenous Execution."

[WP40] David Keyes, Lois Curfman McInnes, Carol Woodward, William Gropp, and Eric Myra, "A Considered Approach to Multiphysics Problems at the Exascale: Coupled Until Proven Decoupled."

[WP41] Richard Vuduc, "Can algorithms inform architecture?."

[WP42] Karen Devine, Sivasankaran Rajamanickam, and Erik Boman, "Combinatorial Scientific Computing for Exascale Systems and Applications."

[WP43] Janine Bennett, C. Seshadhri, Ali Pinar, and David Thompson, "Sublinear Algorithms for In-situ and In-transit Data Analysis at the Extreme-Scale."

[WP44] Erik G. Boman, Karen Devine, Siva Rajamanickam, and Eric Phipps, "Partitioning and Sparse Computations at Exascale."

[WP45] Jeremiah Willcock and Andrew Lumsdaine, "Minimizing Exascale Memory Bandwidth Usage through Sparse Matrix Compression."

[WP46] Michael Minion, "Efficient Temporal Integration at The Exascale."

[WP47] Xiaoye Li, "Hierarchical Algorithms with Reduced Communication/Synchronization."

[WP48] Aydin Buluc, "Applied Mathematics for Data Analysis in the Exascale."

[WP49] Chandrika Kamath, "Mining Massive Complex Datasets on Emerging Architectures."

[WP50] Peter Graf, Michael Sprague, and Ray Grout, "Coupling of hierarchical multiphysics models and other mathematical issues in extreme scale computing."

[WP51] Xiao-Chuan Cai, "Parallel Space-time Methods for Time Dependent Optimization Problems."

[WP52] Bijan Mohammadi, "Principal angles between subspaces and reduced order modelling accuracy in scalable robust optimization."

[WP53] Rick Archibald, Constantinescu Emil, Katherine Evans, Hal Finkel, Terry Haut, Boyanna Norris, Mathew Norman, Adrin Sandu, Miroslav Stonyanov, Mayya Tokman, Beth Wingate, and Yulong Xing, "Resilient, Communication-Reducing, and Adaptive Time Stepping to Accelerate Exascale Scientific Applications."

[WP54] Todd Munson, Paul Hovland, and Stefan Wild, "Adaptive Precision Arithmetic for Reducing Memory and Increasing Computation."

[WP55] Peer-Timo Bremer, "In-Situ Data Transformations to Enable Exascale Analysis."

[WP56] Michael Heroux, "Toward Resilient Algorithms and Applications."

[WP57] Uli Ruede, "New Mathematics for Exascale Computational Science?."

[WP58] Frederic Nataf, "Innovation for Exascale computing."

[WP59] Erik G. Boman, "Randomized and Asynchronous Algorithms for Large Linear Systems."

[WP60] Ivan Yotov and Paolo Zunino, "The interplay between numerical solvers and computational geometry for exascale computing."

[WP61] Jude Zhu, “Physical Properties Preserving Algorithms.”

[WP62] Paul Constantine, David Gleich, and Joseph Nichols, “MapReduce’s Role in Mathematics for Extreme-Scale Computing.”

[WP63] F. Battaglia, Christopher Beattie, Serkan Gugercin, Christopher John Roy, Eric de Sturler, Adrian Sandu, Layne Watson, Mehrdad Shahnam, Madhava Syamlal, Dave Engel, and Xin Sun, “Essentially Local Approaches to Exascale UQ.”

[WP64] Deepak Rajan, “Solving stochastic optimization problems using exascale-ready algorithms.”

[WP65] Iain Duff and Serge Gratton, “Development of mathematical models for Exascale and beyond.”

[WP66] Alex Povitsky and Shunliu Zhao, “Combined continuum and molecular methodology for micro- and nano- scale flows through filters.”

[WP67] John Gilbert, “Algorithmic Primitives for Exascale Computational Discrete Mathematics.”

[WP68] James Demmel, “Avoiding Communication at Exascale.”

[WP69] Greg Bronevetsky, “Management of Variable Accuracy for Application Optimization.”

[WP70] Greg Bronevetsky, “Towards Comprehensive Resilience for Scientific Applications.”

[WP71] Misun Min and Paul Fischer, “Exascale Computing Challenges for Energy Harvesting Systems.”

[WP72] Jed Brown, “Vectorization, communication aggregation, and reuse in stochastic and temporal dimensions.”

[WP73] Dmitry Karpeev, Olle Heinonen, and Juan de Pablo, “Algorithms for Exascale Computational Mesoscience.”

[WP74] Adrian Sandu, “Fusing Information from Models and Measurements at the Exascale.”

[WPA1] Mark F. Adams, Jed Brown, and Matt Knepley, “Low-communication techniques for extreme-scale multilevel solvers.”

Workshop Participants

Name	*Organization*
Rick Archibald	Oak Ridge National Laboratory
Jeffrey Banks	Lawrence Livermore National Laboratory
John Bell	Lawrence Berkeley National Laboratory
Erik Boman	Sandia National Laboratories
Peer-Timo Bremer	Lawrence Livermore National Laboratory/ University of Utah
Greg Bronevetsky	Lawrence Livermore National Laboratory
Jed Brown	Argonne National Laboratory
Peter Brune	Argonne National Laboratory
Xiao-Chuan Cai	University of Colorado Boulder
Luis Chacon	Los Alamos National Laboratory
Edmond Chow	Georgia Institute of Technology
Paul Constantine	Colorado School of Mines
Eric de Sturler	Virginia Tech
Bert Debusschere	Sandia National Laboratories
James Demmel	UC Berkeley
Karen Devine	Sandia National Laboratories
Lori Diachin	Lawrence Livermore National Laboratory
Jack Dongarra	University of Tennessee
Robert Falgout	Lawrence Livermore National Laboratory
David Gleich	Purdue University
William Gropp	University of Illinois
Serkan Gugercin	Virginia Tech.
Terry Haut	Los Alamos National Laboratory
William Harrod	U.S. Department of Energy
William Henshaw	Lawrence Livermore National Laboratory
Michael Heroux	Sandia National Laboratories
Jeffrey Hittinger	Lawrence Livermore National Laboratory
Thuc Hoang	U.S. Department of Energy
Paul Hovland	Argonne National Laboratory
Xiaozhe Hu	The Pennsylvania State University
David Keyes	KAUST and Columbia University
Matthew Knepley	University of Chicago
Dana Knoll	Los Alamos National Laboratory
Tzanio Kolev	Lawrence Livermore National Laboratory
Alice Koniges	Lawrence Berkeley National Laboratory
Randall Laviolette	Office of Science, U.S. DOE
Steven Lee	DOE ASCR
Sven Leyffer	Argonne National Laboratory
Xiaoye Li	Lawrence Berkeley National Laboratory
Michael Mascagni	Florida State University
Lois McInnes	MCS Division, Argonne National Laboratory
Michael Minion	Lawrence Berkeley National Lab/Stanford University
Todd Munson	Argonne National Laboratory
Thomas Ndousse-Fetter	U.S. Department of Energy
Esmond Ng	Lawrence Berkeley National Laboratory
Joseph Nichols	University of Minnesota

Matthew Norman	Oak Ridge National Laboratory
Assad Oberai	Rensselaer Polytechnic Institute
Karen Pao	DOE Office of Science
Eric Phipps	Sandia National Laboratories
Ali Pinar	Sandia National Laboratories
Sivasankaran Rajamanickam	Sandia National Laboratories
Philippe Ricoux	TOTAL/EESI
Ulrich Ruede	University Erlangen
Sonia Sachs	U.S. Department of Energy
Onkar Sahni	Rensselaer Polytechnic Institute
Adrian Sandu	Virginia Tech
Mark Shephard	Rensselaer Polytechnic Institute
Miroslav Stoyanov	Oak Ridge National Laboratory
Michela Taufer	University of Delaware
Brian Van Straalen	Lawrence Berkeley National Laboratory
Wim Vanroose	University of Antwerp, Belgium
Xavier Vasseur	CERFACS
Richard Vuduc	Georgia Institute of Technology
Qiqi Wang	MIT
Clayton Webster	Oak Ridge National Lab
Stefan Wild	Argonne National Lab
Jeremiah Willcock	Indiana University
Beth Wingate	Los Alamos National Laboratory
Carol Woodward	Lawrence Livermore National Laboratory
Jinchao Xu	Pennsylvania State University
Ulrike Yang	Lawrence Livermore National Laboratory
Duan Zhang	Los Alamos National Laboratory
Yu Zhuang	Texas Tech University

Workshop Agenda

Wednesday, 21 August 2013

Time	Speaker	Paper	Title
08:30 - 09:00	Karen Pao		Opening Remarks
09:00 - 10:00	**Session 1** *(Chair: Paul Hovland; Discussion begins at 9:30)*		
	Lori Diachin	19	Scalable Mesh-Based Simulation Workflows are required for Exascale Computers
	Sven Leyffer	15	Mixed-Integer PDE-Constrained Optimization
	Siva Rajamanickam	42	Combinatorial Scientific Computing for Exascale Systems and Applications
10:00 - 10:30	**Break**		
10:30 - 12:00	**Session 2** *(Chair: Luis Chacon; Discussion begins at 11:30)*		
	Lois Curfman McInnes	40	A Considered Approach to Multiphysics Problems at the Exascale: Coupled Until Proven Decoupled
	Assad Oberai	27	Adaptive Multiscale Predictions at Exascale
	Duan Zhang	28	Tree-like Processor Architecture for Multi-scale, Multi-material and Multi-physics Computation Based on the Material Point Method
	Jeff Banks	14	The design and development of modular and adaptive algorithms
	Dana Knoll	25	Moment-based scale-bridging algorithms for multiphysics kinetic simulations at the exascale
	Matt Knepley	73	Algorithms for Exascale Computational Mesoscience
12:00 - 13:30	**Lunch**		
13:30 - 15:00	**Session 3** *(Chair: John Bell; Discussion begins at 14:30)*		
	Michael Minion	46	Efficient Temporal Integration at The Exascale
	Qiqi Wang	37	Scalable Parallel-in-Time Simulation of Extreme-Scale Chaotic Systems
	Beth Wingate	1	Computing Highly Oscillatory Systems at the Exascale
	Xiao-Chuan Cai	51	Parallel Space-time Methods for Time Dependent Optimization Problems
	Rick Archibald	53	Resilient, Communication-Reducing, and Adaptive Time Stepping to Accelerate Exascale Scientific Applications
	Jed Brown	72	Vectorization, communication aggregation, and reuse in stochastic and temporal dimensions
15:00 - 15:30	**Break**		
15:30 - 17:00	**Session 4** *(Chair: Stefan Wild; Discussion begins at 16:30)*		
	Paul Constantine	62	MapReduce's Role in Mathematics for Extreme-Scale Computing
	Ali Pinar	43	Sublinear Algorithms for In-situ and In-transit Data Analysis at the Extreme-Scale
	Peer-Timo Bremer	55	In-Situ Data Transformations to Enable Exascale Analysis
	Eric de Sturler	63	Essentially Local Approaches to Exascale UQ
	Adrian Sandu	74	Fusing Information from Models and Measurements at the Exascale
	Eric Phipps	13	Realizing Exascale Performance for Uncertainty Quantification

Thursday, 22 August 2013

Time	Speaker	Paper	Title
08:30 - 10:00	**Session 5** *(Chair: Rob Falgout; Discussion begins at 09:30)*		
	Xavier Vasseur	65	Development of mathematical models for Exascale and beyond
	Peter Brune	11	Nonlinear Solver Algorithms at the Exascale: Rethinking the Full Linearization Bottlenecks
	Wim Vanroose	22	Hiding global communication latency and increasing the arithmetic intensity in extreme-scale Krylov solvers
	Xiaoye Li	47	Hierarchical Algorithms with Reduced Communication/Synchronization
	Ulrike Yang	9	Multilevel Methods Combine All Properties that are Essential for Fast and Efficient Performance at the Extreme Scale
	James Demmel	68	Avoiding Communication at Exascale
10:00 - 10:30	**Break**		
10:30 - 12:00	**Session 6** *(Chair: Jack Dongarra; Discussion begins at 11:30)*		
	Uli Reude	57	New Mathematics for Exascale Computational Science?
	Jinchao Xu	21	Design and Analysis of Fault-tolerant Multilevel Iterative Algorithms
	Bert Debusschere	32	Probabilistic Approaches for Communication Avoidance and Resilience in Exascale Simulations
	Tzanio Kolev	17	Scalable multi-physics simulations will require new discretization and numerical methods research
	Alice Koniges	36	Consideration of Asynchronous Algorithms for Particle-Grid Simulations
	Erik Boman	59	Randomized and Asynchronous Algorithms for Large Linear Systems
12:00 - 13:30	**Lunch**		
13:30 - 14:40	**Session 7** *(Chair: Esmond Ng; Discussion begins at 14:10)*		
	Richard Vuduc	41	Can algorithms inform architecture?
	Michael Mascagni	31	Monte Carlo Methods Are Perfectly Situated to Enable Exascale Scientific Computing
	Jeremiah Willcock	45	Minimizing Exascale Memory Bandwidth Usage through Sparse Matrix Compression
	Todd Munson	54	Adaptive Precision Arithmetic for Reducing Memory and Increasing Computation
14:40 - 15:00	**Break**		
15:00 - 16:00	**Session 8** *(Chair: Jeff Hittinger; Discussion begins at 15:30)*		
	Greg Bronevetsky	70	Towards Comprehensive Resilience for Scientific Applications
	Brian van Straalen	39	Heterogeneous Computing
	Michael Heroux	56	Toward Resilient Algorithms and Applications
16:00 - 16:30	Karen Pao		Concluding Remarks