DIGITAL MATERIAL: A FRAMEWORK FOR MULTISCALE MODELING OF DEFECTS IN SOLIDS

C. R. MYERS,[1] S.R. ARWADE,[2] E. IESULAURO,[2] P.A. WAWRZYNEK,[2] M. GRIGORIU,[2]
A.R. INGRAFFEA,[2] P.R. DAWSON,[3] M.P. MILLER,[3] AND J.P. SETHNA[4]
[1]Cornell Theory Center
[2]School of Civil and Environmental Engineering
[3]Sibley School of Mechanical and Aerospace Engineering
[4]Laboratory of Atomic and Solid State Physics
Cornell University, Ithaca, NY 14853

ABSTRACT

We are designing and developing a novel software environment – the Digital Material – to support materials modeling across many length and time scales, in order to develop improved descriptions of material structure and evolution. Software support is required for high-performance numerical kernels, lightweight infrastructures for prototyping, steering and analysis, information transfer across scales, coupling of disparate simulation modules, and collaboration between a multidisciplinary collection of researchers.

INTRODUCTION

The need for multiscale modeling of materials

There is widespread recognition – fueled to a considerable extent by the continued growth of available computing power – that improved descriptions of material structure and response will result from the development of models which incorporate information from smaller length and time scales into material models at larger scales. Efforts typically focus either on summarizing small scale processes for use in larger scale models, explicitly coupling material models at multiple length scales, or some combination of both approaches. Extracting meaningful information, however, from complex, multiscale, multidisciplinary models is a nontrivial task, requiring new types of software support for scientific computing. As part of several coupled scientific and engineering research efforts, we have begun to design and develop a flexible and expressive software environment for multiscale materials modeling, dubbed *Digital Material*. In this paper we highlight some of its key design and implementation features.

Digital Material makes extensive use of a number of software engineering techniques which have, to date, not seen widespread use in computational science and engineering. These include, most importantly, the use of interpreted scripting languages to coordinate, steer and integrate a complex suite of simulation and analysis tools, and the use of object-oriented design principles and related techniques such as "design patterns" to introduce needed flexibility into the modeling process.

Some challenges of multiscale modeling of materials

Much of material behavior – especially in engineering systems of interest to us, such as polycrystalline metallic alloys used in structural applications – is driven by the complex interaction and evolution of defect structures at mesoscales. Therefore, a suitable modeling framework must provide support for simulating and understanding complex, three-dimensional dynamical process of many degrees of freedom. Since models of such phenomena are still not well-developed, there must be support for prototyping, validating and interpreting different sorts of models. Support for rapid prototyping cannot impinge, however, on the ever-present need for computational performance.

In the course of research, scientists and engineers naturally describe material entities and structures at various length scales, including electrons, atoms, vacancies, dislocations, voids, lattices, grains, grain boundaries, twin boundaries, dislocation structures, cracks, surfaces, and precipitates. It is therefore important that there be modeling and programming support to describe those structures. It is equally important that researchers be able to develop simulation codes in which the underlying representation of such structures can be altered; this will support not only the process of validating reduced-order models but also the need for run-time adaptivity, whereby certain material features can be simulated more faithfully by explicit treatment of small scales if such a need arises.

Because the nature of material structures and mathematical models differ considerably from scale to scale, it is important that multiscale descriptions be able to represent material information consistently across scales. We have found it useful to recall that various descriptions of material structures are typically approximations to the fully $3N-$dimensional configuration space described by N atoms in a system, or the $4N-$dimensional configuration spacetime describing the evolution of the system. (We are choosing for now not to explicitly include electronic degrees-of-freedom in such a description, but instead to summarize them those into attributes attached to atomic degrees-of-freedom.) One could presumably, with sufficient computer resources, reconstruct the structure and evolution of all the larger scale features in a material (grains, grain boundaries, precipitates, cracks, etc.) from the underlying atomic configuration. Fortunately, most atomic degrees of freedom are not actually required to achieve an accurate description of material structure and response, because of the strongly localized nature of deformation in materials. Coarse-grained theories (such as linear elasticity) often adequately describe the deformations of many atoms in a material.

Such descriptions involve various *approximations* to the underlying atomic configuration space. A major challenge, therefore, is to develop an *optimal* approximation – one that balances *fidelity* as provided by smaller scale structures and processes with *efficiency* as provided by reduced-order descriptions at larger scales. Since there is no one situation or set of resources to dictate which is *the* optimal tradeoff, one must actually develop a systematic framework for deciding and computing an optimal approximation based on questions of interest and resources at hand. In some respects, we envision something that is a generalization of some of the ideas at the core of the quasicontinuum method.[1] But instead of instantiating only atomic degrees-of-freedom to resolve small scale structure, we envision the need to instantiate a hierarchy of structures (atoms, dislocations, dislocation tangles, voids, cracks, etc.) at various scales in order to optimally approximate the underlying configuration space.

DIGITAL MATERIAL

The Digital Material framework is being built to address these and other challenges. At its heart, Digital Material is intended as a problem-solving environment that will support the development of reduced-order descriptions of material structure and response for use at larger scales, the explicit coupling of material models at multiple scales, and the extraction of information from complex, coupled, dynamical processes in materials.

Digital Material as a Material Representation

The material and defect structures that constitute the focus of materials research must be available to the programmer as computational building blocks, to be accessed, manipulated and combined in arbitary ways. We may have, for example, a geometric model of a grain boundary triple junction (derived, say, from experimental measurements), which we decide to model atomistically. To do so, we would like to be able to compose the description of the junction with a recipe for instantiating atomic degrees-of-freedom that is independent of the details of the junction. Should we decide to represent the junction by a collection of dislocations rather than a collection of atoms, we should be able to replace the atomistic generator with one that creates dislocations.

Developing models of material and defect structures so that they can be treated as computational building blocks is naturally achieved through the use of object-oriented techniques. Constructing flexible couplings between such objects, so that we have a composable suite of simulation modules rather than a single simulation program, requires further design considerations, which fall under the category of "design patterns".[2] Design patterns are a class of software engineering techniques which describe the interactions between collections of objects, in such a way as to encapsulate aspects of a problem which need to be varied. Our vision of multiscale modeling involves varying, potentially at run time, a number of aspects, including: the internal representation of material structures, the assembly of objects which are combined to define a specific material "sample", the identity of collective objects (i.e., whether an object is modeled as a collection of small-scale structures for use at one scale or as a primitive object for use at a larger scale). Perhaps most importantly, design patterns provide natural support for the core of our modeling architecture: the separation of geometric structures from attributes attached to those structures and from the tools, models and other external entities that act to query or modify those structures.

Digital Material as an Interactive Problem-Solving Environment

The field of scientific programming continues to unfold, exploiting new tools and methodologies to allow for more expressive computational modeling. Press and Teukolsky[3] have identified the need for a middle ground between traditional low-level programming languages that are "close to the machine" and more recent "total environments" that purport to provide support for all aspects of scientific computing (but which have dubious performance for large-scale numerical computing). In addition, interactivity in programming and analysis environments is increasingly recognized as a key to unraveling complex phenomena.[4]

Although the analogy is not entirely correct, it would not be inappropriate to describe our Digital Material as something akin to a "Matlab for Materials". Matlab is very successful at providing high-level support for research in linear algebra, allowing computational

experimentation by freeing the programmer from low-level details. Unfortunately, we find that Matlab is less well-suited for problems outside the realm of linear algebra, so we have decided on a different core.

The central element of Digital Material which supports high-level interactive programmability as well as "glue" to integrate many different modules is Python, an interpreted, object-oriented programming language.[5] In addition to being a high-level programming language, Python is also an extension language capable of dynamically loading compiled modules written in more traditional programming languages. By allowing the incorporation of low-level extension modules, Python is able to provide high-level language support for rapid prototyping and interactive exploration, while not sacrificing computational performance in time-critical sections of code. Python is freely available and runs on all platforms of interest; for scientific and numerical computing, modules are available to support array manipulations, linear algebra, FFTs, geometric operations, interpolation, polynomial fits, automatic differentiation, plotting, GUI development, 3D visualization, image processing, Internet programming, and database access (to name a few). The process of incorporating one's own simulation or analysis code into the larger Python environment is vastly facilitated by the SWIG package[6], developed by David Beazley originally in support of the SPaSM materials modeling project at LANL.[7] SWIG is a nonintrusive "interface and wrapper generator" which automatically produces the "glue code" that enables Python to communicate with C or C++. SWIG also ingeniously allows for the development of object-oriented structure on top of legacy codes which do not possess such structure.

With tools such as these, we will be able to build a flexible modeling, simulation and analysis environment, by mixing and matching components as needed. Rapid prototyping can be done very effectively within Python, with time-critical pieces migrated down to a compiled language as performance becomes an issue. Ultimately, we will be able to use heuristic insights gained through interactive exploration to automate highly complex processes, such as adaptive creation of more refined small-scale models in response to particular types of material deformation.

INITIAL APPLICATIONS

Because our assembled group is diverse in its background and focus, research is ongoing on a number of problems at a number of different length scales. We are in the process now of developing new simulation capabilities, and integrating existing capabilities into a more unified framework. Initial applications include: determining the structure, energetics and mobility of elementary defects in atomic lattices in two dimensions (with J. Tomasi and X. Chen), integrating components for three-dimensional finite-element simulations of crack propagation (as part of a related project with K. Pingali, S. Vavasis, P. Chew, D. Schneider, P. Stoghill and N. Chrisochoides), and investigating the nature of texture and misorientation and their roles in microstructural failure processes such as intergranular fracture. We will breifly describe below some of the ongoing research in this last area.

Texture, misorientation, dislocation structures and fracture in polycrystals

The large-scale plastic deformation of polycrystalline metals can involve the formation of intricate dislocation structures and, at larger scales, the development of preferred crystalline orientations, or texture. Such structures and patterns are intellectually fascinating

in their own right, and can also have a profound impact on material properties, e.g., on the fracture toughness of a metal prone to intergranular cracking. We are working to couple simulations of large-scale plasticity and texture evolution[8] to codes focused on predicting crack growth.[9] Increasingly sophisticated experimental tools – such as Electron Backscatter Diffraction Patterns (EBSP) – are also available to probe lattice orientations and misorientations on mesoscales. An example of EBSP orientation data for an aluminum alloy are shown in Figure 1. Data such as these will enable us to characterize the statistical properties of orientation distributions, generate synthetic material samples for simulation, and test hypotheses concerning the detailed form of dislocation structures that form to delineate grain and subgrain boundaries.

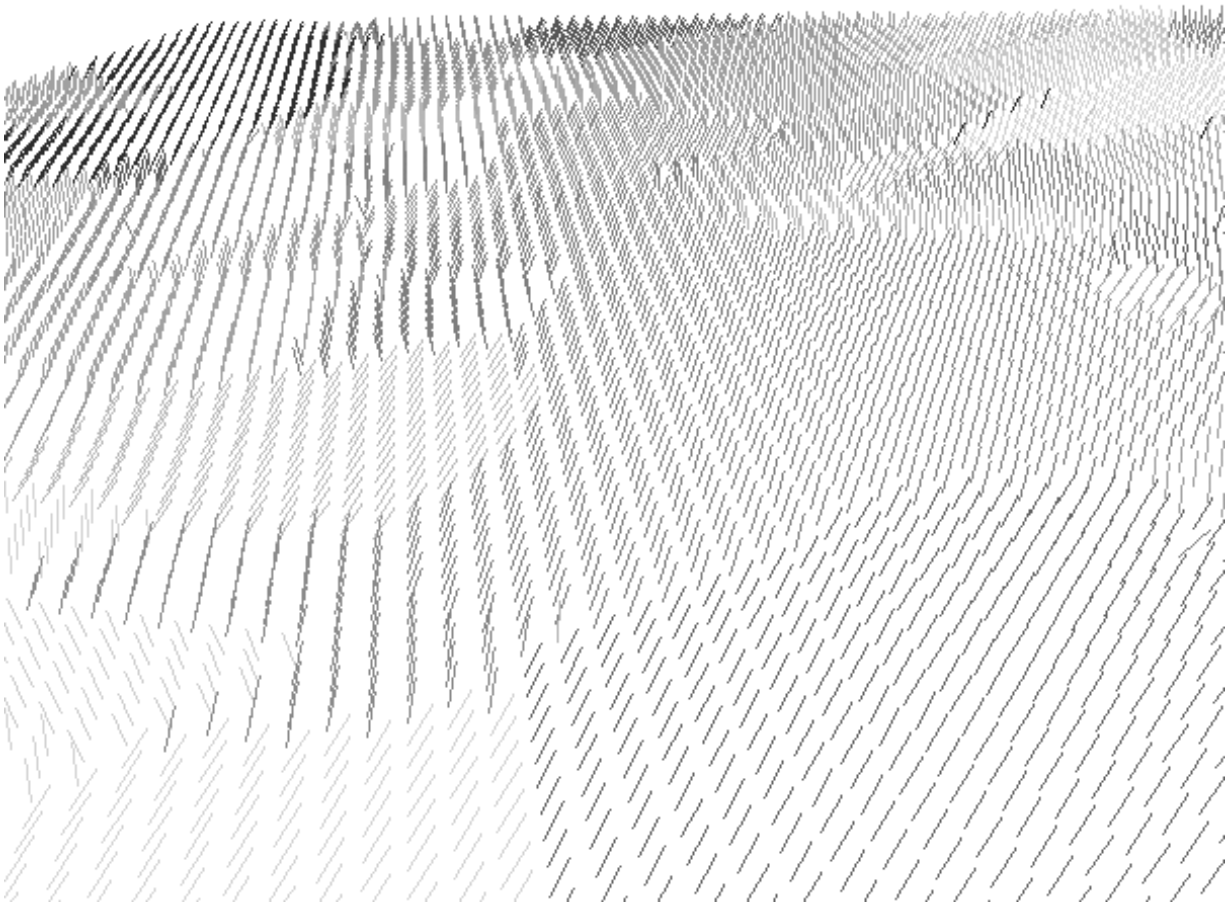

Figure 1: Experimental lattice orientation data on the surface of a polycrystalline aluminum alloy, as determined by Electron Backscatter Diffraction. Local lattice orientation is measured on a regular triangular grid of side length 5 microns; a section of a full scan of size $540\mu m \times 540\mu m$ is shown. (Line directions and grayscales represent orientation axis and angle, respectively, in a Rodrigues axis-angle parameterization.) Misorientation between scan locations indicates the presence of grain or subgrain boundaries. By incorporating such data into the Digital Material environment, we can validate models of dislocation structure formation and texture evolution, and develop statistical models to generate synthetic material samples for simulation.

CONCLUSIONS

The peculiar demands of multiscale modeling of materials require new strategies for simulation, analysis, interrogation and collaboration. The complex, dynamic adaptivity of materials needs to be addressed with computational tools that are equally complex, dynamic and adaptive. Multiscale modeling environments must provide systematic programming support for information transfer across scales. Lightweight scripting and extension environments like Python can effectively balance the conflicting requirements of high performance and rapid prototyping. All of these elements are conspiring to bring life to Digital Material.

ACKNOWLEDGMENTS

This work is supported by the NSF and the AFOSR under grants NSF 9873214 and AFOSR F49620-98-1-0401.

References

[1] E.B. Tadmor, M. Ortiz, and R. Phillips, Phil. Mag. A **73**, no.6, 1529 (1996).

[2] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison Wesley Longman, Reading, MA (1995).

[3] W.H. Press and S.A. Teukolsky, Computers in Physics **11** (5), 416 (1997).

[4] P.F. Dubois, Computers in Physics **8** (1), 70 (1994).

[5] The Python Home Page, http://www.python.org.

[6] SWIG: Simplified Wrapper and Interface Generator, http://www.swig.org.

[7] D.M. Beazley and P.S. Lomdahl, Proceedings of Supercomputing '96, Pittsburgh, PA (1996).

[8] A. Kumar and P.R. Dawson, Comp. Methods Appl. Mech. Engr., in press.

[9] Gray, L., D. Potyondy, E. Lutz, P. Wawrzynek, L. Martha, and A. Ingraffea, Mathematical Models and Methods in Applied Sciences **4(2)**, 179 (1994).