

Genetic Programming for Multiscale Modeling

Kumara Sastry & D. D. Johnson*

*Department of Material Science & Engineering,
Fredrick Seitz Materials Research Laboratory,
University of Illinois at Urbana Champaign, Urbana IL 61801*

David E. Goldberg

*Department of General Engineering,
University of Illinois at Urbana Champaign, Urbana IL 61801*

Pascal Bellon

*Department of Material Science & Engineering,
Fredrick Seitz Materials Research Laboratory,
University of Illinois at Urbana Champaign, Urbana IL 61801*

ABSTRACT

We propose the use of genetic programming (GP)—a genetic algorithm that evolves computer programs—for bridging simulation methods across multiple scales of time and/or length. The effectiveness of genetic programming in multiscale simulation is demonstrated using two illustrative, non-trivial case studies in science and engineering. The first case is multi-timescale materials kinetics modeling, where genetic programming is used to symbolically regress a mapping of all diffusion barriers from only a few calculated ones, thereby avoiding explicit calculation of all the barriers. The GP-regressed barrier function enables use of kinetic Monte Carlo for realistic potentials and simulation of realistic experimental times (seconds). Specifically, a GP regression is applied to vacancy-assisted migration on a surface of a binary alloy and predict the diffusion barriers within 0.1–1% error using 3% (or less) of the barriers. The second case is the development of constitutive relation between macroscopic variables using measured data, where GP is used to evolve both the function form of the constitutive equation as well as the coefficient values. Specifically, GP regression is used for developing a constitutive relation between flow stress and temperature-compensated strain rate based on microstructural characterization for an aluminum alloy AA7055. We not only reproduce a constitutive relation proposed in literature, but also develop a new constitutive equation that fits both low-strain-rate and high-strain-rate data. We hope these disparate example applications exemplify the power of GP for multiscaling at the price, of course, of not knowing physical details at the intermediate scales.

*Address all correspondence to duanej@uiuc.edu

1. INTRODUCTION

In recent years there has been growing interest in developing effective modeling and simulation methods to explain or predict the behavior of many phenomena in science and engineering. Many of these phenomena are inherently multiscale. For example, to simulate realistic nano-layered film growth, we often require systems with sizes in the range of 10–100 nm (simulation cells over one million atoms), and simulation times of about 10–1000 seconds [1]. Thus, there is a significant premium on cost-effective modeling techniques that can simulate physical, chemical, or biological phenomena across multiple scales in both time and space, even at the price of losing information at intermediate scales. Many powerful methods exist that address modeling at single scales and only recently has attention been paid to develop computational methods that span multiple scales of time and space.

One approach is to apply modeling methods of a single scale and couple them by transferring key information from the finer scale to a coarser scale [2]. For example, microstructural information from atomistics can be used to regress a constitutive relation between stress and strain that in turn can be used in a finite-element analysis. An important and often daunting task in this multiscale approach is the development of proper coupling methods. In this paper, we propose *symbolic regression* via genetic programming (GP)—a genetic algorithm that evolves computer programs—as a robust method for coupling modeling methods from different scales. Unlike traditional regression methods, symbolic regression via GP automatically and adaptively develops both the functional form and the coefficient values based on phenomenological behavior at an underlying scale. For example, rather than assuming, say, a Fourier basis (which may prejudice the outcome) and regressing the Fourier coefficients, we allow both function and coefficients to evolve and search for a more optimal function.

We believe that genetic programming-based symbolic regression holds promise in various multiscale areas, for example, modeling multiscale kinetics, obtaining constitutive rules, and finding chemical reaction pathways. Also, as we exemplify with case studies, standard basis-set regression are generally not competitive to GP for fixed accuracy due to the difficulty in choosing appropriate basis functions. To demonstrate our methodology's effectiveness, we discuss the application GP to multiscale modeling of vacancy-assisted migration on (100) surface of phase-separating $\text{Cu}_x\text{Co}_{1-x}$, where local chemical environment around a vacancy result in numerous barriers needed for simulations, and to derive a constitutive relation between flow stress and temperature-compensated strain rate for aluminum alloy AA7055.

This paper is organized as follows. A brief introduction of the mechanisms and workings of genetic programming is presented in the following section. The potential of applying GP for multiscale simulation in science and engineering is discussed in Sec. 3. The effectiveness of symbolic regression via genetic programming in multiscale modeling is demonstrated with the help of two illustrative, but nontrivial examples from two major areas in Sec. 4: (i) evolving a constitutive relation between flow stress and temperature-compensated strain rate for an aluminum alloy, and (ii) multiscale materials kinetics simulation. Finally, we provide summary, conclusions, and possible future directions of the work.

2. GENETIC PROGRAMMING

Genetic programming [3–7] is a genetic algorithm [8–10] that evolves computer programs via the mechanisms of genetics and natural selection. Over the last two decades, GP has been successfully used to solve problems in a wide range of areas from novel analog circuit design to quantum computing that has resulted in sev-

eral reinventions of patented inventions and some patentable new inventions [7]. A typical GP consists of the following components:

Representation: Unlike traditional search methods, genetic algorithms encode the decision variables of a problem into an artificial *chromosome*, e.g., in a binary representation, a solution (chromosome) could be (0101110111) in a 2^{10} solution space. In GP a chromosome is a candidate computer program, which is usually represented by a tree [4], see Fig. 1. However, it should be noted that other representations such as linear codes [11], grammar-based codes [12, 13], and domain-specific representations [7, 14] have also been used.

In tree representations, the internal nodes of a tree are composed of elements from a set of *primitive functions* and the leaf nodes consist of elements from a set of *terminals* as shown in the example of Fig. 1. Both the primitive functions and the terminals are user specified. The primitive functions can be arithmetic functions (e.g., “+”, “-”, “*”, “/”), logical expressions (e.g., “if-then-else,” “and,” “or,” “not”), boolean functions (e.g., “AND,” “OR,” “XOR,” “NOT”), loops (e.g., “while,” “for”), and other program constructs, including user-specified subroutines, or domain-specific functions. The terminals usually consists of independent variables of a problem, constants, and *ephemeral random constants* [4].

We note here briefly that choosing appropriate primitive functions and terminals is one of the key factors influencing GP performance. In some cases, the choice of the primitive functions can be straightforward and might consist of arithmetic functions and a branching operator. In other cases, the primitive functions might include specialized functions from the problem domain. For example, for thermal processes, such as various creep mechanisms, a specialized function, such as $Q/(k_B T)$, can be beneficial. Similarly, for thermal activation process, such as vacancy-assisted climb (which is

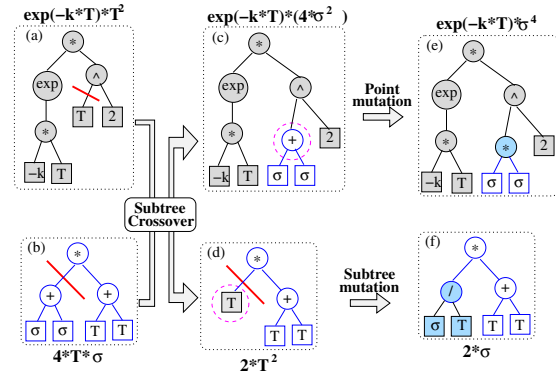


FIGURE 1. Illustration of tree representation, subtree crossover, subtree mutation, and point mutation used in GP.

one of the creep-recovery mechanisms), using $\exp[-Q/(k_B T)]$ as a primitive function might be beneficial in obtaining an optimal solution. Reference [4] suggests selecting primitive functions that are capable of expressing the solution to the problem, and calls this the *sufficiency* property. Reference [15] suggests using “the most powerful useful seeming functions from the problem domain that you can think of.” It should be noted, however, that if one uses advanced GP features like *automatically defined functions* and *architecture altering operations*, then missing, but necessary primitive functions can be automatically co-evolved [6].

Notion of Fitness: To evolve computer programs and to implement natural selection, we need a measure for distinguishing good solutions from bad solutions. This can involve the usual elaborate computer simulation or mathematical model that helps determine what good is (the standard notion of an *objective* function), or it can be as simple as having a human intuitively choose better solutions over worse ones (what we might call a *subjective* function). It can even be an ecologylike process where different digital species *co-evolve* through an intricate mix of competition and cooperation. However it is done, *something* must determine a solution’s relative *fitness to purpose* (or *context*), and what-

ever that is will be used by the genetic program to guide the evolution of future generations. For example, in symbolic regression for fitting some data, the fitness function could be the sum-squared error between the predicted and calculated values. Or, for a traveling salesman problem, perhaps minimum travel distance is most appropriate.

Notion of Population: Unlike traditional search methods genetic algorithms and programming rely on a population of candidate solutions. The population size, which is a user-specified parameter, is one of the important factors affecting the scalability and performance of genetic algorithms. For example, small population sizes might lead to premature convergence and yield substandard solutions. On the other hand, large population sizes lead to unnecessary expenditure of valuable computational time. While population sizing in genetic algorithms has been analytically studied [16, 17], population-sizing models for GP are still in the early developmental stages [18].

Once the problem is encoded in a chromosomal manner and a fitness measure for discriminating good solutions from bad ones has been chosen, we can start to *evolve* solutions to the search problem using the following steps:

1. **Initialization:** The initial population of candidate solutions is usually generated randomly across the search space. However, domain-specific knowledge and other information can be easily incorporated in the generation of initial population. A common initialization scheme used in GP is the *ramped half-and-half* method [4], where trees of different sizes (between user-specified minimum and maximum tree sizes) and shapes are initialized using the *grow* and *full* methods in equal proportion. In a *grow* method, a tree of arbitrary size is generated by selecting terminals or primitive functions with equal probability. In a *full* method a tree of specified size is generated by selecting only primitive functions for the nodes till the tree size approaches a specified size after which only terminals are selected.
2. **Evaluation:** Once the population is initialized or a offspring population is created, the fitness values of the candidate solutions are evaluated.
3. **Selection:** Allocates more copies to solutions with better fitness values and thus imposes the survival-of-the-fittest mechanism on the candidate solutions. The main idea of selection is to *prefer better solutions to worse ones*, and many selection procedures have been proposed to accomplish this idea, including roulette-wheel selection, stochastic universal selection, ranking selection, and tournament selection. One of the population selection procedures is an *s-wise tournament selection* [19], where *s* candidate solutions are randomly chosen and pitted against each other in a tournament. A solution with the best fitness wins. We note, however, that to have a sufficiently robust *gene pool* one must maintain some viable, but not necessarily high-fitness solutions.
4. **Recombination:** Combines bits and pieces of two or more parental solutions to create new, possibly better solutions (i.e., offspring). There are many ways of accomplishing this, and achieving competent performance does depend on getting the recombination mechanism designed properly; but the primary idea to keep in mind is that the offspring under recombination will not be identical to any particular parent and will instead combine parental traits in a novel manner [10]. Competent recombination operators that automatically identify the important traits of parental solutions and effectively exchange have been developed for genetic algorithms [10, 20–22], and recently for genetic programming [23].

Typically, in GP a *subtree-crossover* method

is used [4]. In subtree crossover, a crossover point for each solution is randomly chosen and subtrees below this crossover are swapped to create two new solutions (see Fig. 1).

5. **Mutation:** While recombination operates on two or more parental chromosomes, mutation locally but randomly modifies a solution. Again, there are many variations of mutation, but usually involves one or more changes are made to an individual's trait or traits. By itself mutation represents a "random walk" in the neighborhood of a particular solution [10].

In GP, usually two mutation techniques are used, see Fig. 1: *Subtree mutation*, where a subtree is randomly replaced with another randomly created subtree, and *point mutation* where a node is randomly modified.

6. **Replacement:** The offspring population created by selection, recombination, and mutation replaces the original parental population. Many replacement techniques such as elitist replacement, generation-wise replacement and steady-state replacement methods are used in GP.
7. Repeat steps 2–6 till one or more convergence criteria are met.

2.1. Advanced Genetic Programming Features

The scalability and applicability of GP can be significantly enhanced with the help of one or more of the advanced features [6, 7], two of which are outlined in the following:

Constrained Syntactic Structures: In simple GP, the search space—the possible set of valid combinations of functions and terminals—is usually unconstrained. However, when solving

complex problems, it might be advantageous, or even necessary, to search among a restricted program space. Usually, grammar-based representations such as *strong typing*, are used to restrict the space of valid syntactic structures [7, 24, 25]. One such restriction (or constraint) that is useful for symbolic regression is imposing the requirement that the GP operators create (or promote) only *dimensionally-correct* functions (or programs) [13, 14]. For example, when evolving a model for predicting activation energies, we need to search among only those relations whose dimensions are consistent with energy units. By restricting the GP to evolve only dimensionally-correct programs, the search space can be significantly reduced and the evolutionary process can be accelerated.

Automatically Defined Functions (ADFs): Sometimes it is difficult or even impossible to have prior knowledge of all the important primitive functions that are useful for solving the problem at hand. Even when appropriate primitive functions are used, the solution might contain repeated use of certain highly favorable combinations of functions and terminals. For example, when modeling activation energies from atomistic simulations, a relation representing the attractive and repulsive forces between atom pairs, such as $(r/\sigma)^n - (r/\sigma)^m$, might be repeatedly used. In such cases, GP can automatically evolve highly favorable combinations of functions and terminals and *encapsulate* them into reusable subroutines [5]. These automatically defined functions eliminate the need to "reinvent the wheel" and efficiently utilizes the modularities, symmetries, regularities, and hierarchy of the problem [7]. Moreover, the use of ADFs decreases the possibility of disrupting the important sub-programs via the genetic operators, such as recombination and mutation.

In GP, an ADF is evolved from a population of candidate ADFs in parallel to the main pro-

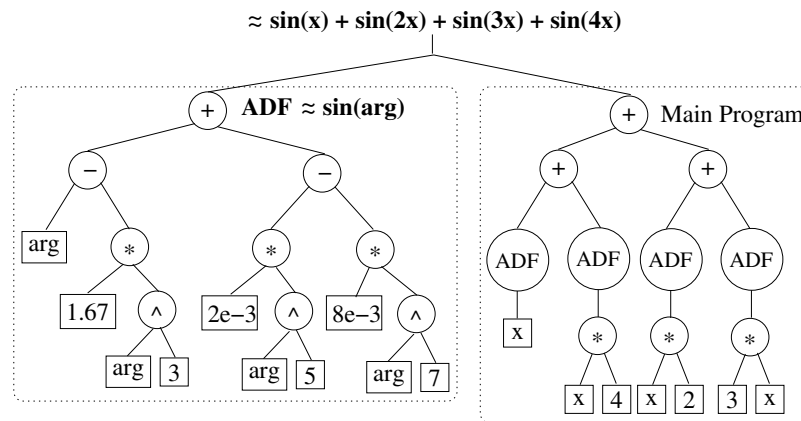


FIGURE 2. Illustration of an ADF. The individual approximates $\sin(x) + \sin(2x) + \sin(3x) + \sin(4x)$ by repeatedly using an ADF, which in turn approximates the \sin function.

gram. That is, every individual comprises of multiple trees, one representing the main program and the others representing the ADFs as shown in Fig. 2. The main program invokes the ADFs as a primitive function with a predefined number of arguments. For example, in Fig. 2, the ADF takes a single arguments. Other program features, such as iterations, loops, recursions, and stores, can also be automatically defined along the lines of an ADF and further details are available elsewhere [5, 26–28]. Furthermore, the structure, content, and the subroutine topologies can also be evolved with the help of *architecture altering operations* such as addition, deletion, and duplication of subroutines and arguments [6].

3. GP-REGRESSION FOR MULTISCALING IN SCIENCE AND ENGINEERING

Although there has been a rise in interest in multiscale as of late [1, 29–43], the very notion of multiscale has been an important part of mathematical physics for some time. Perhaps the most commonplace examples are drawn from the physics of solids, liquids, and gases, where assumptions are made about the molecular behavior of matter and statistical mechanics enables us to derive constitutive relationships that enable us to treat the mechanics

of solids, liquids, and gases in a continuum. These cases are, of course, somewhat special, depending on the microscopic homogeneity of the molecular level. As engineers and scientists want to understand less regular low-level phenomena, multiscale speed-up depends on our ability to derive custom-made constitutive relationships for the special case at hand. No longer can we reliably rely on specialized mathematical tools to bridge the gap, and the two modeling levels being bridged may not be assumed to exist in mathematical closed form. Instead, we must find ways to automatically (i) sparsely sample low-level models, and (ii) derive accurate custom-made constitutive relationships for a higher level using a uniform, competent computational procedure.

We propose symbolic regression via genetic programming as one such effective tool for bridging modeling methods working on different scales. That is, we propose the use of GP to evolve automatically (reduced-order) models between macroscopic variables based on microscopic data (see Fig. 3).¹ Unlike traditional

¹It should be noted that like genetic programming, other machine-learning and data-mining techniques such as artificial neural networks and Bayesian-learning techniques can also be used for multiscale modeling. For example, see Tiley *et al.* [44].

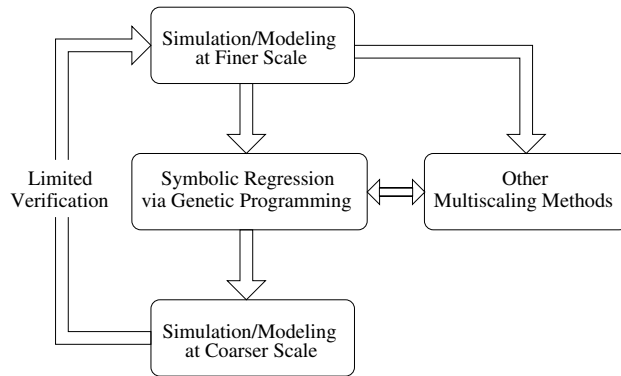


FIGURE 3. Schematic of GP-based approach for multiscale simulation.

regression methods, GP does not require the knowledge of the functional form of the coupling relation. In fact, GP searches for both the functional form (via appropriate choice of primitive functions, terminals, and program-tree structure) and regresses the coefficients in parallel to best fit the data. That is, GP *evolves* important basis functions for regression via appropriate hierarchical combination of primitive functions, and also optimizes the coefficient values simultaneously.

However, similar to other regression methods, the symbolic regression via GP yields a function that “fits” the data and usually does not reveal the underlying physics. Yet the flexibility of genetic programming makes it readily amenable to the incorporation of problem-specific knowledge through the choice of primitive functions, terminals, and via custom design of representation, recombination, and mutation operators. Moreover, the GP can be developed such that only functions are generated that manifestly satisfy dimensional/unit requirements. Other multiscale methods can also be readily and efficiently *hybridized* with GP as in the case of genetic algorithms [45]. Furthermore, genetic programming is inherently parallel making it readily amenable to a variety of parallelization techniques [46], which is a significant advantage over other regression/optimization methods.

4. GP-REGRESSION EXAMPLE MULTISCALING APPLICATIONS

We now exemplify the utility and effectiveness of the GP-approach in multiscale simulations

by applying the methodology to two current areas of interest. These examples constitute a fundamental science problem in multitime-scale material kinetics modeling and an engineering problem in development of constitutive relation between flow stress and strain rate. We also demonstrate that standard basis-set regression are generally not competitive to GP for fixed accuracy due to the difficulty in choosing appropriate basis functions.

4.1. Evolving Constitutive Relations

We demonstrate the capabilities of genetic programming in evolving constitutive relations based on macroscale data arising from microscale effects. One of the ways to transfer information from microscopic analysis onto macroscopic analysis is via constitutive relations. A methodology for automatically discovering not only the important variables, but also the functional form of constitutive relations between them can be very effective. Such reduced-order constitutive relations can then be used in macroscopic analysis via finite-element or finite-difference methods.

Here, we wish to emphasize the effectiveness of GP for this application. Hence, we use measured data [47] and evolve a constitutive relation between flow stress and temperature-compensated strain rate for an aluminum alloy AA7055. Specifically, the constitutive relation is required for the simulation of the hot rolling processes, which is one of the important thermomechanical processes involved in the treatment of aluminum alloys [48]. The effect of temperature, stress, and strain rate on the final product properties and their anisotropy are required to understand the hot rolling process. Furthermore, the material properties are also affected by changes in precipitate distributions, grain structure, and texture [49]. Microstructural characterization can be used to describe the microstructural development with temperature and the evolution of deformation struc-

tures. Moreover, high-temperature compression tests can be performed to high strains to collect data that can correlate material properties with microstructural damage characterization [47].

The objective is to find a constitutive relation between the stress and strain rate to model their correlation based on microstructural characterization. Padilla *et al.* [47] conducted such high-temperature compression tests on AA7055 at temperatures ranging from 340–520°C at two different strain rates of 1 s^{-1} and 10^{-3} s^{-1} . A detailed experimental procedure used in obtaining the stress-strain rate data is given elsewhere [47]. Assuming the power-law relation between stress and strain rate, they obtained the coefficients that best fit the data to give the following constitutive relation:

$$\dot{\epsilon} = A_o \exp\left(\frac{Q_d}{RT}\right) \left(\frac{\sigma}{\mu}\right)^{4.56}, \quad (1)$$

where $\dot{\epsilon}$ is the strain rate, σ is the stress, μ is viscosity, Q_d ($= 125 \text{ kJ/mol}$) is the activation energy for diffusion of zinc in aluminum, R is universal gas constant, T is the temperature, and A_o is an Arrhenius exponent.

We demonstrate the effectiveness of GP in automatically discovering constitutive relations without any problem-specific knowledge (other than the choice of primitive functions and terminals) by applying GP to evolve the stress-strain rate relationship for AA7055. Here we use the function set

$\mathcal{F} = \{+, -, *, /, ^, \exp, \sin\}$ and the terminal set $\mathcal{T} = \{\dot{\epsilon}, T, \exp[1/(RT)], \mathcal{R}\}$. Here \mathcal{R} is a random number generator and in genetic programming parlance called *ephemeral random constant* [4]. We use both T as well as $\exp[1/(RT)]$ as possible temperature-related variables to see if GP can automatically decide between the more likely useful primitive function $\exp[1/(RT)]$ over the less likely T . The output of the candidate program is the ratio of stress and viscosity, i.e., σ/μ .

The fitness of a solution is computed as the absolute error between the predicted and experimental data for σ/μ :

$$f = \frac{1}{M} \sum_{i=1}^M \left| \left(\frac{\sigma}{\mu}\right)_{\text{pred}} - \left(\frac{\sigma}{\mu}\right)_{\text{expt}} \right|, \quad (2)$$

where M is the number of experimental data points. This fitness function is one of the obvious choices for data-fitting problems. Here we note that experimental uncertainties can be incorporated into the fitness function in a straightforward manner. For example, a Gaussian noise (or other models of the uncertainty) can be added to the fitness function, which would prevent the GP from over-fitting to the data. Alternatively, if any of GP-regressed data is within the error-bar of its corresponding measured data, we can set the error to zero.

First, we start by obtaining constitutive relations between stress-strain rate using only low-strain-rate data. That is, we only used experimental data obtained for strain rate of 10^{-3} s^{-1} . This was because the power-law relation described by Padilla *et al.* [47] was fit only to the low-strain-rate data. Therefore, we wanted to verify if GP could find a similar relation as that of Eq. (1). We expect that this linear regression should be trivially reproduced, albeit by a highly nontrivial approach. We find that GP does indeed discover a stress-strain rate relation that is in agreement with the constitutive relation of Padilla *et al.* [47] and other power-law relations used to describe creep in metals [50, 51].

$$\dot{\epsilon} = c_o \exp\left(\frac{1}{RT}\right) \left(\frac{\sigma}{\mu}\right)^{4.55}, \quad (3)$$

where $c_o \approx A_o \exp(Q_d)$ is a constant.

We ran 10 independent runs of GP (which took wall time of 13–41 sec per GP run on a 1.67 GHz AMD Athlon XP workstation with two users) and in all 10 runs we obtained very

similar relations as above, suggesting that the above expression might be an optimum. Furthermore, GP was able to select the appropriate form for incorporating the effect of temperature in the constitutive relation. The comparison of the constitutive relation developed by GP and that developed by [47] [Eq. (1)] to experimental data is shown in Fig. 4.

Now, more importantly, we use GP-regression and fit both the low-strain-rate and high-strain-rate data and evolved the constitutive relation. We were interested in finding out if GP can identify a single relation to fit both the data sets. Note that the GP does not have any knowledge of the two different sets of data, that is, high- or low-strain rates. As far as GP is concerned, there is no qualitative difference between high-strain-rate and low-strain-rate data. We used the same set of functions and terminals as in the previous case and obtained the following constitutive relation for the

$$\dot{\epsilon} = \frac{c_o \exp\left(\frac{1}{RT}\right)}{g\left[\dot{\epsilon}, \exp\left(\frac{1}{RT}\right)\right]} \left(\frac{\sigma}{\mu}\right)^4 \left[1 - \frac{\sigma}{\mu}\right], \quad (4)$$

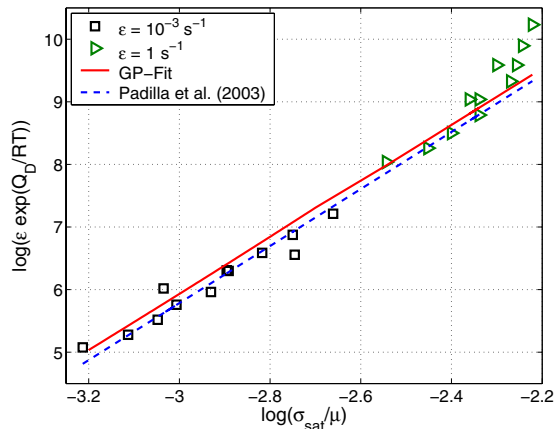


FIGURE 4. Constitutive relation developed by GP [Eq. (3)] and used by Padilla *et al.* [47] [Eq. (1)] between stress and strain rate for accounting variations in microstructural deformation. Only low-strain-rate data is used for the symbolic regression via GP.

where the denominator g is a complex mathematical expression:

$$g = 10.3 \left[1 - 3 \left(\frac{\sigma}{\mu}\right) \sqrt{6 + \sqrt{\left|A_o e^{-\frac{Q_D}{RT}} - 2\frac{\sigma}{\mu}\right|}} \cdot \left\{ 1 - 3\frac{\sigma}{\mu} \cdot A_o e^{-\frac{Q_D}{RT}} \cdot \left(1 - A_o^2 e^{-\frac{2Q_D}{RT}} + 2\frac{\sigma}{\mu}\right) \right\} \right]. \quad (5)$$

The behavior and functionality of g is explained in the later paragraphs. Equation (4) indicates a competition between the fourth-order power-law $[(\sigma/\mu)^4]$ and fifth-order power-law $[(\sigma/\mu)^5]$ in stress. Such power-law behavior is suggestive of possible competing mechanisms taking place during the microstructural deformation process. Furthermore, three out of ten independent GP runs (which took wall time of 34.4–58.2 sec per GP run on a 1.67 GHz AMD Athlon XP workstation with 2 users) yield similar constitutive relation as Eq. (4). The rest of the seven GP runs to gave either a fourth-order power-law or a fifth-order power-law equation still highlighting the competition between the two.

Notably, a fifth-order power-law represents creep in metals [51]. However, it is unclear as to the physical mechanism represented by the fourth-order power law. Nonetheless, GP regression appears to have consistently identified a competing set of power-law-mechanisms; one which has clear physical meaning (creep), while the other remains unknown, but as discussed below, yield cross-over from low-strain high-strain rate.

Comparison of the constitutive relation [Eq. (4)] with experimental data is shown in Fig. 5. The results show that the constitutive relation developed by GP does indeed agree with experimental data. Furthermore, there is a noticeable kink incorporated in Eq. (4) at the transition between low- and high-strain-rate data points. This indicates that GP implicitly identified the transition point between

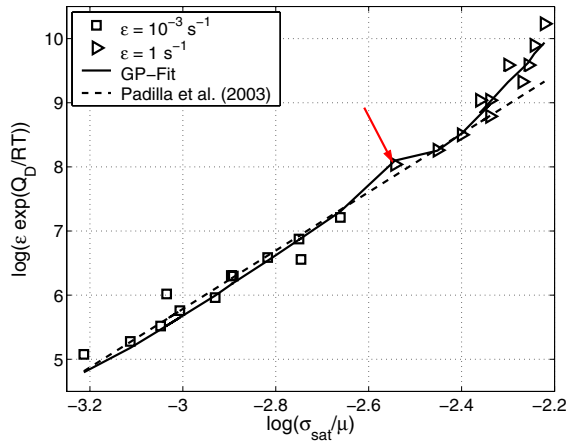


FIGURE 5. Constitutive relation developed by GP [Eq. (4)] and used by Padilla *et al.* [47] [Eq. (1)] between stress and strain rate for accounting variations in microstructural deformation. Both low-strain-rate and high-strain-rate data are used for the symbolic regression via GP. The arrow indicates a crossover between low-strain-rate and high-strain-rate data.

high and low-strain-rate data. After some simplification and analysis of the denominator g in Eq. (4), we found that the kink was represented in the denominator. This indicates that GP was able to identify a missing variable and compensate for the missing variable by a step function as shown in Fig. 6. It is important to note that this transition is well established. For example, a hyperbolic function is often used to specify the transition between low-strain-rate and high-strain-rate data [51], which approximates a step-function-like behavior in analytic models.

Our results in this section demonstrate the effectiveness of genetic programming in regressing constitutive relation between key macroscopic variables using microscopic information. Such a reduced-order model can be highly effective in multiscaling not only in space coordinates, but also in time coordinates. The constitutive relations developed in this paper can be readily used with finite-element or finite-difference methods along the lines of *et al.* [47] to simulate the hot rolling process used in the treatment of aluminum alloys. However, we

stress that while the GP provided a better constitutive law that incorporates strain-rate effects, it does not reveal the underlying physics, in this case a competing creep mechanism and as to yet unknown mechanism.

4.2. Multiscale Material Kinetics Simulation

In the previous section, we demonstrated the use of genetic programming for regressing constitutive relation between macroscopic variables based on data arising out of microscale effects. In this section, we discuss the use of genetic programming in multiscale material kinetics modeling. We begin with a brief discussion of current simulation methods in the following paragraphs.

Molecular dynamics (MD) [52–54] is extensively used for kinetic modeling of materials. Yet, MD methods are limited to nanoseconds of real time and hence fail to model directly many processes. Recently, several approaches were proposed for multiscaling [1, 34–43]. Methods such as temperature-accelerated dynamics (TAD) [35] provide significant acceleration of MD, but they still fall 3–6 orders of magnitude short of real processing times. These methods assume that transition-state theory applies, and concentrate only on infrequent events. An alternative approach to bridge timescales uses kinetic Monte Carlo (KMC) [55–57] combined with MD [1], which relied on an *a priori* list of events (so-called table look-up). A table of possible events is commonly comprised of atomic jumps, but collective motions (or off-lattice jumps) may need to be considered depending on the problem e.g., see [39]. Tabulating activation energies from a (possibly innumerable) list of events is a serious limitation. For example, multicomponent alloys have an impossibly large set of barriers, due to configurational dependence, making their tabulation impractical, especially from first-principles. An alternative approach is calculating energies “on-the-fly” [38, 58], but it too has a serious limitation

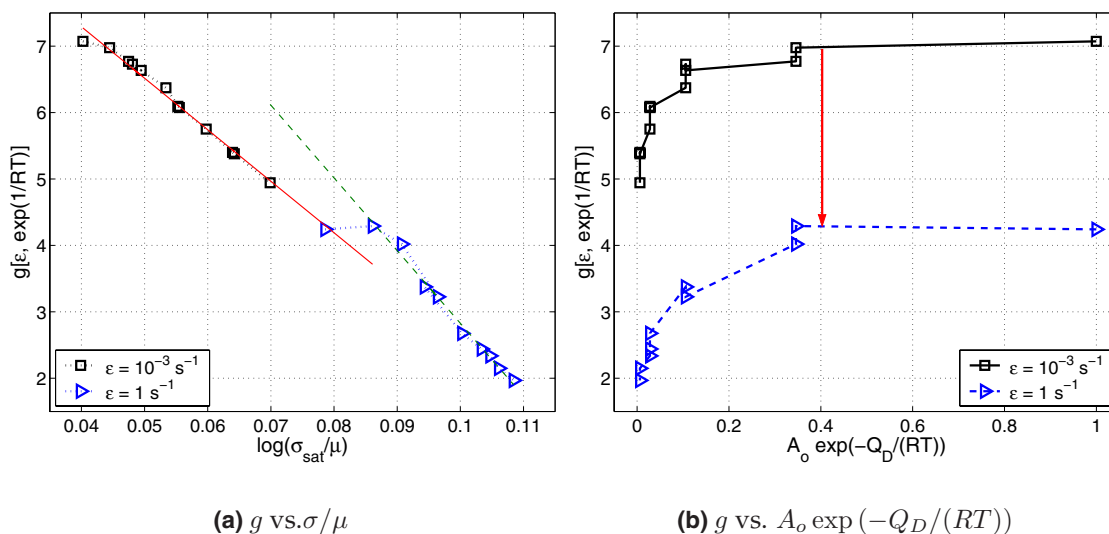


FIGURE 6. The effect of the denominator $g\epsilon, \exp[1/(RT)]$ for different strain rates and temperature. The results show that the denominator is a complex way of expressing a simple step function. The step function, which was automatically discovered by GP, represents the transition between low-strain-rate and high-strain-rate data points.

of time restriction [58]. For example, encountering infrequent events on-the-fly might take unrealistically long time.

However, alternatively we could generate the look-up table by *machine learning* the barrier energies and, rather than actually calculating all the barriers, we could calculate only a few and regress a function that accurately reproduces other barriers. The barrier function than is just an in-line function. It should be noted that table look-up KMC requires the knowledge of all the barriers and even then takes $\mathcal{O}(n \log n)$ time to lookup the barrier, where n is the total number of events. Similarly, on-the-fly KMC also requires encountering different barriers during the course of the simulation. On the other hand, if we *machine learn* the barriers based on few configurations, we can potentially predict barriers for other, yet unencountered, configurations. Furthermore, the barrier function is just an in-line function call taking only milliseconds (usually independent of the number of barriers/events). We call such a KMC method augmented with machine-learned bar-

rier function as symbolically regressed table KMC (sr-KMC).

4.2.1. Symbolically-Regressed Table KMC (sr-KMC)

Therefore, in order to avoid the need or expense of explicit calculation of all activation barriers—frequent or infrequent—and thereby facilitate an effective hybridization of MD and KMC for multiscale dynamics modeling, we utilize genetic programming to symbolically regress the entire PES from a limited set of calculated points on the PES. An accurate GP-regressed PES extends the KMC paradigm to machine learn the look-up table for increasing complexity and providing simulation over experimentally relevant time frames, not possible from either table look-up KMC, on-the-fly KMC, or MD simulations.

To demonstrate our effectiveness of srKMC, we apply GP to a non-trivial case of vacancy-assisted migration on (100) surface of phase-separating $\text{Cu}_x\text{Co}_{1-x}$. Although there are mil-

lions of configurations, only the environmental atoms locally around vacancy and migrating atom significantly influence the barrier energies. We refer to these as the *active* configurations. The results show that GP predicts barriers within 0.1–1% error using calculated barriers of less than 3% of the total *active* configurations. For alloys, this technique can be combined with a local cluster expansion technique [59] that reduces the explicit barrier calculations to $\sim 0.3\%$ of the *active* configurations. Our initial results hold promise to enable the use of KMC (even with realistic potentials) for increased problem complexity with a scale-up of simulation time over MD.

Here we use the function set $\mathcal{F} = \{+, -, *, /, ^, \exp, \sin\}$ and the terminal set $\mathcal{T} = \{\vec{x}, \mathcal{R}\}$, where \vec{x} is a vector representing the *active* alloy configuration, and \mathcal{R} is an ephemeral random constant [4]. Since we use GP for predicting the barriers, a tree represents a PES-prediction function that takes a configuration and ephemeral constants as inputs and returns the barrier for that configuration as output.

For the fitness measure f , we calculate the barriers $\{\Delta E_{\text{calc}}(\vec{x}_1), \dots, \Delta E_{\text{calc}}(\vec{x}_M)\}$ for M random configurations $\{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_M\}$. These configurations are used as inputs to the tree and the barriers $\{\Delta E_{\text{pred}}(\vec{x}_1), \dots, \Delta E_{\text{pred}}(\vec{x}_M)\}$ are predicted. The fitness is then computed as a weighted average of the absolute error between the predicted and calculated barriers:

$$f = \frac{1}{M} \sum_{i=1}^M w_i |\Delta E_{\text{pred}}(\vec{x}_i) - \Delta E_{\text{calc}}(\vec{x}_i)| \quad (6)$$

with $w_i = |\Delta E_{\text{calc}}|^{-1}$, which gives preference to accurately predicting lower energy (most significant) events.

In this particular application (surface vacancy-assisted migration), the fitness f maintains agreement with a (perhaps randomly picked) set of calculated diffusion barriers. The weights favor the lowest-energy (more favorable) pathways through the chemical “landscapes” of the diffusion atoms.

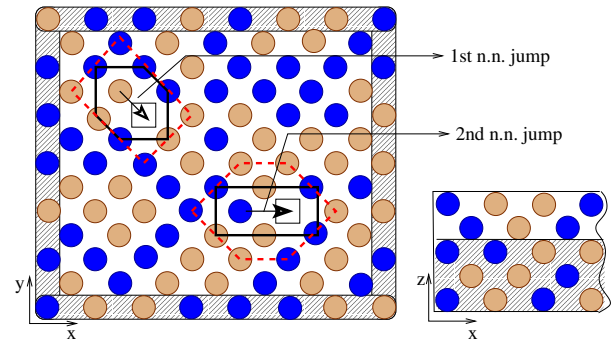


FIGURE 7. Sketch of simulation cell for vacancy-assisted migration on (100)-surface of an fcc binary alloy. Atoms in all but the bottom layers and the boundary can fully relax. The solid (dashed) lines around the migrating atom and vacancy represent 1st (2nd) n.n. environmental atoms.

4.2.2. Case Study: Vacancy-Assisted Migration on (100) $\text{Cu}_x\text{Co}_{1-x}$

To demonstrate the effectiveness of genetic programming, we consider the prediction of diffusion barriers for vacancy-assisted migration on (100) surface of phase-separating $\text{Cu}_x\text{Co}_{1-x}$. The test system consists of five layers with 100–625 atoms in each layer (see Fig. 7). The bottom three layers are held fixed to their bulk bond distances, while the top layers are either held fixed (as a test) or fully relaxed via MD. We consider only first and second nearest-neighbor (n.n.) jumps, along with 1st (as a test) and 2nd n.n. environmental atoms in the active configuration, as shown in Fig. 7. Table 1 gives the number of active configurations when 1st and 2nd n.n. environments are considered.

For simplicity, we model the atomic interactions with the Morse potential [60]. To validate interactions, we model vacancy-assisted migration on (100)-surface of Cu and consider only first n.n. jumps. The predicted barrier for n.n. vacancy jumps with fully relaxed lattice in Cu is 0.39 eV, agreeing with 0.42 ± 0.08 (0.47 ± 0.05) from *ab initio* (EAM) [61] calculations. However, we emphasize that the underlying atomistic method and associated potentials do not impact the GP whatsoever. The GP

Table 1. Number of active configurations for 1st and 2nd n.n. jumps, and for 1st and 2nd n.n. active atoms

	1 st n.n. jumps	2 nd n.n. jumps
1 st n.n. active configurations	128	128
2 nd n.n. active configurations	2048	8192
Total configurations	$\gg 2^{100}$	$\gg 2^{100}$

only requires barrier information from which to learn. They could be measured barriers, calculated barriers, or realistically, reliable barriers, so that we get quantitative results.

We now consider the PES regression via GP for vacancy-assisted migration on (100)-surface of $\text{Cu}_x\text{Co}_{1-x}$. The input to the PES regression (i.e., prediction) function, $\vec{x} = \{x_j\}$ is a binary-encoded vector sequence, where $x_j = 0$ (1) represents a Cu (Co) atom. For simplicity, we begin by considering only seven 1st n.n. environmental atoms yielding 128 *active* configurations. About 20 (i.e., 16%) different active configurations are randomly chosen and their barriers are computed using the conjugate-gradient method and are used in the GP fitness function [see Eq. 6]. The barriers predicted by GP for the

relaxed configurations are compared to the exact values in Fig. 8. We note that the prediction error for rigid lattice case ($0.4 \pm 0.04\%$) is significantly less than that for relaxed lattice case ($2.8 \pm 0.08\%$). Due to the weighting used in the fitness function, GP predicts barriers for most significant events more accurately than for less-significant (higher-energy) events.

Figure 8 also compares the barriers predicted by GP to those predicted by a least-squares fit quadratic polynomial, showing clearly its inadequacy for alloys generally. Furthermore, while GP requires only 16%, the quadratic (cubic) polynomial fit needs 27% (78%) of the barriers. In limited cases, such as dilute $\text{Fe}_{1-x}\text{Cu}_x$, the barriers can be predicted via a simple polynomial fit [62].

To test the scalability of GP with *active* configuration size, we consider the 2nd n.n. jumps and 1st and 2nd n.n. environmental atoms in the *active* configuration. As shown in Table 1, there are a total of 8192 configurations. The energies predicted by GP are compared with direct calculations in Fig. 9, along with the error. The GP predicts the barriers for most significant events with less than 0.1% error by fitting to energies from only 3% (i.e., 256/8192) of the active configurations. The average relative error is given by

$$\bar{\epsilon}_{\text{rel}} = \frac{100}{N'_{\text{cfgs}}} \sum_{i=1}^{N'_{\text{cfgs}}} \left| \frac{\Delta E_{\text{pred}}(\vec{x}_i) - \Delta E_{\text{calc}}(\vec{x}_i)}{\Delta E_{\text{calc}}(\vec{x}_i)} \right|, \quad (7)$$

where N'_{cfgs} is the number of configurations within the desired energy range. A cubic polynomial fit requires energies for $\sim 6\%$ of the configurations, predicting the barriers with 2.5% error for the most significant events.

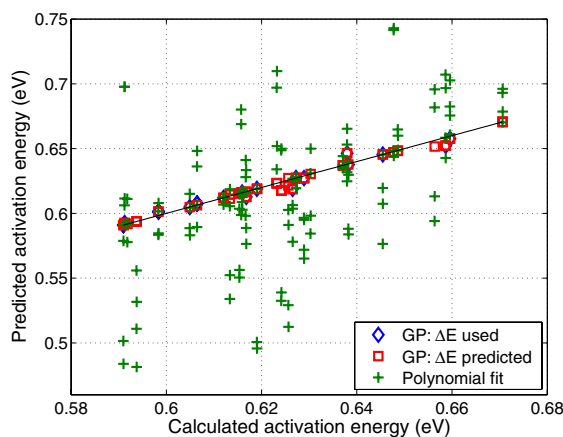


FIGURE 8. Activation energies (in eV) predicted by regression. GP (circles) and a quadratic polynomial (crosses) are compared to the calculated barriers for 1st n.n. jumps on (100)-surface of $\text{Cu}_{0.5}\text{Co}_{0.5}$ for relaxed lattices. For ease of visualization, only first n.n. environments are considered in the active configuration. The line is a guide for the eye.

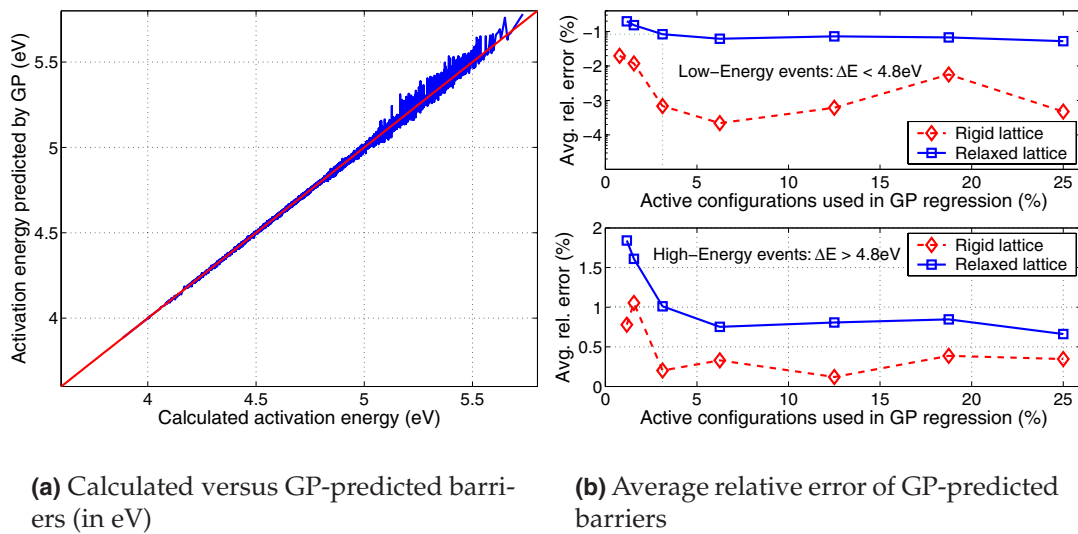


FIGURE 9. (a) Calculated vs. GP-predicted barriers (in eV) for 2nd n.n. jumps on relaxed (100)-surface of $\text{Cu}_{0.5}\text{Co}_{0.5}$ active configurations up to 2nd n.n. (b) GP predicts the barriers with 0.1% (1%) error for most-significant (less-significant) events with $\Delta E < 4.8 \text{ eV}$ ($\Delta E > 4.8 \text{ eV}$) from only 3% of active configurations.

The results shown in Figs. 8 and 9 clearly demonstrate the effectiveness of GP in predicting the potential energy surface, with high accuracy and little information. As expected, we find that the GP performance does not depend on the potentials (linear or non-linear) used. For example, the same results are found for the linear potentials of Lennard-Jones, Double-Yukawa, or Morse potentials. Non-linear potentials, such as EAM and first-principle, would also show this because GP does not depend on which method was used to calculate the barriers. Furthermore, even though the function form of the potentials (linear or non-linear) are significantly different, the resulting barrier heights are usually very similar (as we observed for the case of pure Cu). We also find that the GP performance is independent of the configuration set that is used in calculating the fitness function, the order in which they are used, and the mechanism used to convert the configuration into a vector of inputs. Differences in activation-energy scale on the PES prediction via GP is also negligible. That is, even though the barriers for the 1st and 2nd n.n. jumps differ by an order of magnitude,

GP predicts the barriers with similar accuracy. Moreover, for more complex, cooperative effects, such as island diffusion via surface dislocations [63], sr-KMC can be interfaced with pattern-recognition methods [64]. For long-range fields (e.g., elastic fields from coherent interfaces, such as multilayers or precipitates), a description based solely on local configurations may have to be extended, say, with phase field methods.

The potential time enhancements by coupling a GP-regressed PES with KMC, i.e., sr-KMC, are simple to estimate. With event occurrence following a Poisson distribution, the real time in KMC is given by [55, 65]

$$\tau_r = \sum_{j=1}^{N_{\text{KMC}}} \frac{-\ln(U)}{\sum_{i=1}^{N_{\text{cfgs}}} \nu_o \exp[-\beta(\Delta E(\vec{x}_i))]}, \quad (8)$$

where N_{KMC} is the number of Monte Carlo steps, $U \in (0, 1]$ is a uniform random variable, N_{cfgs} is the number of active configurations, and ν_o is $\approx 27 \times 10^{12} \text{ Hz}$ for Cu [61].

Using sr-KMC we obtain, per time step of KMC relative to MD (assuming an MD time-step of 10^{-15} s), 9 orders of magnitude increase

in *simulated* time over MD at 300 K, 4 orders at 650 K, and 2.3 orders at 1000 K. From the example given, we showed a direct CPU gain of 100 over table look-up methods. Additionally, each time step of sr-KMC requires only 10^{-3} CPU-seconds, as opposed to MD or on-the-fly KMC, which require seconds (empirical potentials) to hours (quantum methods).

To summarize, potential energy surface prediction using symbolic regression via GP holds promise as an efficient tool for multiscaling in dynamics. The GP approach avoids the need or expense of calculating the entire potential-energy surface, is highly accurate, and leads to significant scale-up in simulation time and a reduction in CPU time over molecular dynamics. We have shown on a nontrivial example of vacancy-assisted migration on a surface of $\text{Cu}_x\text{Co}_{1-x}$ that GP predicts all barriers with 0.1–1% error from calculations for only 3% of active configurations, allowing seconds of simulation time. For alloy problems, the number of direct barrier calculations can further be reduced by over an order of magnitude by hybridizing GP with cluster expansion methods [59].

5. SUMMARY AND CONCLUSIONS

In this paper we proposed symbolic regression via genetic programming (GP) as a robust method for bridging simulation methods across multiple scales. Unlike traditional regression methods, symbolic regression via GP adaptively *evolves* both the functional relation as well as regression constants for transferring key information based on modeling at an underlying scale to a higher scale. We demonstrated the applicability of GP in multiscale simulation with the help of two nontrivial examples from science (vacancy-assisted surface migration in alloys—multitimescaling) and engineering (deriving a constitutive relation between flow stress and temperature-compensated strain rate for aluminum alloy AA7055—multispatial scaling).

In the multitimescaling example, GP-based symbolic regression is used predict the diffu-

sion barriers from only a few calculated points, thereby avoiding explicit calculation of the entire PES. The GP-based regression predicts the barriers within 0.1–1% error using 3% (or less) of the barrier calculations and thereby enables use of kinetic Monte Carlo for increased realistic timescales and real systems.

In the latter example, we sought the development of constitutive equation between flow-stress and temperature compensated strain-rate based on microstructural characterization for an aluminum alloy. In other words, GP was used to automatically identify the relationship between macroscopic variables based on microscopic effects. The GP-based approach not only reproduced a constitutive relation proposed in literature for low-strain-rate data, but also developed a new constitutive equation that fit both low-strain-rate and high-strain-rate data, identifying a crossover between the two. It remains to be seen whether it in fact describes additional strain-rates to which it was not fit.

We believe that GP-based symbolic regression holds promise in various multiscaling areas, including finding chemical reaction pathways, which we are currently investigating. The present results indicate that GP-based symbolic regression is an effective and promising tool for multiscale. Moreover, the flexibility of GP makes it readily amenable to hybridization with other multiscaling methods leading to enhanced scalability and applicability to more complex problems.

ACKNOWLEDGMENTS

This work was sponsored by the National Science Foundation under ITR Grant No. DMR-99-76550 at Materials Computation Center, and ITR Grant No. DMR-0121695 at CPSD; the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under Grant No. F49620-03-1-0129; and the Department of Energy through the Fredrick Seitz Materials Research Lab under Grant No. DEFG02-

91ER45439 at UIUC. The U.S. Government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the National Science Foundation, or the U.S. Government.

REFERENCES

- Jacobsen, J., Cooper, B. H., and Sethna, J. P. Simulations of energetic beam deposition: From picoseconds to seconds. *Phys. Rev. B* 58(23):15847–15865, 1998.
- Picu, R. C. Foreword: Multiscale computational materials science—Linking discrete and continuum models. *International Journal for Multiscale Computational Engineering* 1(1):vii–viii, 2003.
- Koza, J. R. Hierarchical genetic algorithms operation on populations of computer programs. *Proc. of 11th Int. Joint Conference on Artificial Intelligence*, 1:768–774, 1989.
- Koza, J. R. *Genetic programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA, 1992.
- Koza, J. R. *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge, MA, 1994.
- Koza, J. R., Bennett III, F. H., Andre, D., and Keane, M. A. *Genetic Programming III: Darwinian Invention and Problem Solving*. Morgan Kaufmann, San Francisco, CA, 1999.
- Koza, J. R., Keane, M. A., Streeter, M. J., Mydlowec, W., Yu, J., and Lanza, G. *Genetic programming IV: Routine Human-Competitive Machine Intelligence*. Kluwer Academic Publishers, Boston, MA, 2003.
- Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: University of Michigan Press.
- Goldberg, D. E. *Genetic algorithms in search optimization and machine learning*. Addison-Wesley, Reading, MA, 1989.
- Goldberg, D. E. *Design of innovation: Lessons from and for competent genetic algorithms*. Kluwer Academic Publishers, Boston, MA, 2002.
- Banzhaf, W., Nordin, P., Keller, R., Francone, F. *Genetic Programming—An Introduction on the Automatic Evolution of Computer Programs and Its Applications*. Morgan Kaufmann, San Francisco, CA.
- Ryan, C., Collins, J., and O’Neill, M. Grammatical evolution: Evolving computer programs for an arbitrary language. *Proc. of EuroGP Conf.*, 1391:83–96. (LNCS), 1998.
- Ratle, A., and Sebag, M. Grammar-guided genetic programming and dimensional consistency: Application to non-parametric identification in mechanics. *Applied Soft Computing* 1:105–118, 2001.
- Babovic, V., and Keijzer, M. Genetic programming as a model induction engine. *Journal of Hydroinformatics* 2(1):35–60, 2000.
- Kinnear, K. E. A perspective on the work in this book. In *Advances in Genetic Programming*, Kinnear, K. E., (ed.), MIT Press, Cambridge, MA, 3–19, 1996.
- Goldberg, D. E., Deb, K., and Clark, J. H. Genetic algorithms, noise, and the sizing of populations. *Complex Systems* 6:333–362, 1992. (Also IlliGAL Report No. 91010).
- Harik, G., Cantú-Paz, E., Goldberg, D. E., and Miller, B. L. The gambler’s ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation* 7(3):231–253, 1999. (Also IlliGAL Report No. 96004).
- Sastry, K., O’Reilly, U.-M., and Goldberg, D. Building-block supply in genetic programming. In *Genetic Programming Theory and Practice*, Riolo, R., & Worzel, B., Eds., Kluwer Academic Publishers, Boston, MA, 155–172, 2003. (Also IlliGAL Report No. 2003012).
- Goldberg, D. E., Korb, B., and Deb, K. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems* 3(5):493–530, 1989.
- Goldberg, D. E., Korb, B., and Deb, K. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems* 3(5):493–530, 1989. (Also IlliGAL Report No. 89003).
- Pelikan, M., and Goldberg, D. E. Escaping hierarchical traps with competent genetic algorithms. *Proc. of the Genetic and Evolutionary Computation Conf.*, 511–518, 2001. (Also IlliGAL Report No. 2000020).
- Pelikan, M. Bayesian optimization algorithm: From single level to hierarchy. Doctoral dissertation, University of Illinois at Urbana-Champaign, Urbana, IL, 2002. (Also IlliGAL Report No. 2002023).

23. Sastry, K., and Goldberg, D. Probabilistic model building and competent genetic programming. In *Genetic Programming Theory and Practice*, Riolo, R., & Worzel, B. Eds., Kluwer Academic Publishers. Boston, MA, 205–220, 2003. (Also IlliGAL Report No. 2003013).
24. Montana, D. J. Strongly typed genetic algorithm. *Evolutionary Computation* 3(2):199–230, 1995. (Also BBN Technical Report No. 7866).
25. Haynes, T. D., Schoenfeld, D. A., and Wainwright, R. L. Type inheritance in strongly typed genetic programming. In *Advances in Genetic Programming 2* Angeline, P. J., & Kinnear, K. E., Eds., MIT Press, Cambridge, MA, 359–376, 1996.
26. Angeline, P. J. Genetic programming and emergent intelligence. In *Advances in Genetic Programming*, Kinnear, K. E., Ed., MIT Press, Cambridge, MA, 75–98, 1996.
27. Kinnear, K. E. Alternatives in automatic function definition: A comparison of performance. In *Advances in Genetic Programming*, Kinnear, K. E. (ed.), MIT Press, Cambridge, MA, 119–141, 1996.
28. Rosca, J. P., and Ballard, D. H. Discovery of subroutines in genetic programming. In *Advances in Genetic Programming 2* Angeline, P. J., & Kinnear, K. E., Eds., MIT Press, Cambridge, MA, 177–202, 1996.
29. Grujicic, M., Cao, G., and Joseph, P. F. Multiscale modeling of deformation and fracture of polycrystalline lamellar γ -TiAl+ α_2 -Ti₃Al alloys. *International Journal for Multiscale Computational Engineering* 1(1):1–21, 2003.
30. Picu, R. C. A nonlocal formulation of rubber elasticity. *International Journal for Multiscale Computational Engineering* 1(1):23–32, 2003.
31. Fish, J., and Schwab, C. Towards constitutive models based on atomistics. *International Journal for Multiscale Computational Engineering* 1(1):43–56, 2003.
32. Mukherjee, T., Fedder, G., Ramaswamy, D., and White, J. Emerging simulation approaches for micromachined devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Devices* 19(12):1572–1589, 2000.
33. Chen, Y. Model order reduction for nonlinear systems. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 1999.
34. Voter, A. F. Hyperdynamics: Accelerated molecular dynamics of infrequent events. *Phys. Rev. Lett.* 78(20):3908–3911, 1997.
35. Voter, A. F. Parallel replica method for dynamics of infrequent events. *Phys. Rev. B* 57(22):R13985–R13988, 1998.
36. Voter, A. F., Montalenti, F., and Germann, T. C. Extending the time scale in atomistic simulation of materials. *Annu. Rev. Mater. Res.* 32:321–346, 2002.
37. Diaz De La Rubia, T., *et al.* Self-decay-induced damage production and micro-structure evolution in fcc metals: An atomic-scale computer simulation approach. *J. Comp.-Aided Mat. Design* 5:243–264, 1998.
38. Henkelman, G., and Jónsson, H. A dimer method for finding saddle points on high dimensional potential surfaces using only first derivatives. *J. Chem. Phys.* 111(15):7010–7022, 1999; *ibid.* 113:9978, 2000; *ibid.* 115:9657, 2001.
39. Sørensen, M. R., and Voter, A. F. Temperature-accelerated dynamics for simulation of infrequent events. *J. Chem. Phys.* 112(21):9599–9606, 2000.
40. Cai, W., Kalos, M. H., de Koning, M., Bulatov, V. V. Importance sampling of rare transition events in Markov processes. *Phys. Rev. E* 66(4):046703, 2002.
41. Steiner, M. M., and Genilloud, P.-A. Simple bias potential for boosting molecular dynamics with the hyperdynamics scheme. *Phys. Rev. B* 57(17):10236–10239, 1998.
42. Barkema, G. T., and Mousseau, N. Event-based relaxation of continuous disordered systems. *Phys. Rev. Lett.* 77(21):4358–4361, 1996.
43. Barkema, G. T., and Mousseau, N. The activation-relaxation technique: An efficient algorithm for sampling energy landscapes. *Comput. Mat. Sc.* 20:285–292, 2001.
44. Tiley, J., Banerjee, R., Searles, T., Kar, S., and Fraser, H. Modeling the relationships between microstructural parameters and tensile properties in Ti₆Al₄V using neural networks and fuzzy-logic models. *Titanium 2003: Proc. of 10th World Conf. on Titanium*, (to published), 2004.
45. Sinha, A., and Goldberg, D. E. A survey of hybrid genetic and evolutionary algorithms, IlliGAL Report No. 2003004, University of Illinois at Urbana-Champaign, General Engineering Dep., Urbana, IL, 2003.
46. Cantú-Paz, E. *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Pub, Boston, MA, 2002.
47. Padilla, H. A., Harnish, S. F., Gore, B. E., Beau-

- doin, A. J., Dantzig, J. A., Robertson, I. M., and Weiland, H. High temperature deformation and hot rolling of AA7055. *Proc. of the 1st Int. Symp. on Metallurgical Modeling of Aluminum Alloys*, 1–8, 2003.
48. Beaudoin, A., and Cassada, W. Investigation of texture and fracture toughness in AA7050 plate. *Proc of the TMS Annual Spring Meeting*, 1998.
 49. Deshpande, N. Relationship between fracture toughness, fracture path, and microstructure of 7050 aluminum alloy: Part I. Quantitative characterization. *Metal Transactions A* **29A**:1191–1201, 1998.
 50. Mukherjee, A. K. High-temperature creep. In *Treatise on Materials Science and Technology*, Vol. 6, Asenault, R. J., Ed., Academic Press, New York, 164–221, 1975.
 51. Kassner, M. E., and Pérez-Prado, M.-T. Five-power-law creep in single phase metals and alloys. *Progress in Materials Science* **45**:1–102, 2000.
 52. Alder, B. J., and Wainwright, T. E. Studies in molecular dynamics. I. General method. *Journal of Chemical Physics* **32**:459–466, 1959.
 53. Alder, B. J., and Wainwright, T. E. Studies in molecular dynamics. II. Behaviour of a small number of elastic spheres. *Journal of Chemical Physics* **33**:1439–1451, 1960.
 54. Allen, M. P., & Tildesley, D. J. *Computer Simulation of Liquids*. Clarendon Press, Oxford, 1989.
 55. Binder, K. (ed.) *Monte Carlo methods in statistical physics*. Springer, Berlin 1986.
 56. Frenkel, D., and Smit, B. *Understanding Molecular Simulation: From Algorithms to Applications*. Academic Press. San Deigo, CA 1996.
 57. Landau, D., and Binder, K. *A Guide to Monte Carlo Simulations in Statistical Physics*. Cambridge University Press, Cambridge, MA, 2000.
 58. Bocquet, J. L. On-the-fly evaluation of diffusional parameters during a Monte Carlo simulation of diffusion in alloys: A challenge. *Defect and Diffusion Forum* **203–205**:81–112, 2002.
 59. Van der Ven, A., and Ceder, G. First-principles theory of ionic diffusion with nondilute carriers. *Phys. Rev. B* **64**(18):184307, 2001.
 60. Girifalco, L., and Weizer, V. Application of the Morse potential function to cubic metals. *Phys. Rev.* **114**(3):687–690, 1959.
 61. Boisvert, G., and Lewis, L. Self-diffusion of adatoms, dimers, and vacancies on Cu(100). *Phys. Rev. B* **56**(12):7643–7655, 1997.
 62. Bouar, Y. L., and Soisson, F. Kinetic pathways from embedded-atom-method potentials: Influence of the activation barriers. *Phys. Rev. B* **65**(9):094103, 2002.
 63. Hamilton, J. C., Daw, M. S., and Foiles, S. M. Dislocation mechanism for island diffusion on fcc (111) surfaces. *Phys. Rev. Lett.* **74**(14):2760–2763, 1995.
 64. Rahman, T., Private communication.
 65. Fichtorn, K. A., and Weinberg, W. H. Theoretical foundations of dynamical Monte Carlo simulations. *J. Chem. Phys.* **95**(2):1090–1096, 1991.