Global Optimization of Parameters in the Reactive Force Field ReaxFF for SiOH

Henrik R. Larsson,[a] Adri C. T. van Duin,[b] and Bernd Hartke*[a]

We have used unbiased global optimization to fit a reactive force field to a given set of reference data. Specifically, we have employed genetic algorithms (GA) to fit ReaxFF to SiOH data, using an in-house GA code that is parallelized across reference data items via the message-passing interface (MPI). Details of GA tuning turn-ed out to be far less important for global optimization efficiency than using suitable ranges within which the parameters are varied. To establish these ranges, either prior knowledge can be used or successive stages of GA optimizations, each building upon the best parameter vectors and ranges found in the previous stage. We have finally arrive-ed at optimized force fields with smaller error measures than those published previously. Hence, this optimization approach will contribute to converting force-field fitting from a specialist task to an everyday commodity, even for the more difficult case of reactive force fields. © 2013 Wiley Periodicals, Inc.

DOI: 10.1002/jcc.23382

Introduction

Chemistry deals with reactions between molecules. The inter- and intramolecular forces governing these reactions result from the electrostatic interactions between the atomic nuclei and the electrons, and from the fundamentally quantum-mechanical nature of the latter. Frequently, it is possible to obviate the need for an explicitly quantum-mechanical treatment by modeling interatomic forces with empirical potentials or force fields. This incurs savings in computer time of about six orders of magnitude, which makes it possible to simulate large-scale systems with billions of atoms[1] or somewhat smaller systems for very long times, currently up to the millisecond regime[2,3] for protein folding, that is, covering 10^{12} time steps. In biochemistry simulations, nonreactive force fields are employed almost exclusively. Nevertheless, various reliable reactive force fields are also in common use in other areas of chemistry.[4–9] Several of them are specific for certain systems or atoms.[10–13] With ReaxFF, a rather general reactive force field has become available in the last decade.[14] After the first parametrization for hydrocarbons,[14] ReaxFF parameters have been produced for various other systems, ranging from inorganic materials[15] via various organo-catalytic transition-metal systems[16,17] to pure metals.[18] Hence, ReaxFF is not limited to certain classes of compounds, and it can handle both periodic and molecular systems. The character of the force-field implementation is kept modular, that is, there should exist only one set of parameters for each atom which can be used in combination with parameter sets of other atoms. Despite the greater computational cost of a more complicated force field like ReaxFF, it was also shown that it is amenable to large-scale simulations.[19]

However, there is no known way to extract analytically accurate force fields from the molecular Schrödinger equation. Additionally, the accuracy requirements of computational chemistry are very strict. Therefore, the standard approach is to fit force fields to reference data, which can be an arbitrary mixture of experimental data and *ab initio* quantum-chemical calculation results.

As fitting empirical force fields to reference data via variation of the force-field parameters is a high-dimensional and nonseparable optimization problem with multiple minima, standard gradient-based local search techniques are of little help. Deterministic global optimization techniques[20,21] are available but face severe practical difficulties in high-dimensional search spaces and for computationally expensive objective functions. Unfortunately, both characteristics are typical for the present problem. Viable alternatives are nondeterministic search heuristics, for example, genetic algorithms (GA),[22,23] or more generally evolutionary algorithms (EA). They offer comparatively very fast access to very good solutions, at the expense of failing to guarantee global convergence. This is ideal for the present task, as there is no need to find the global minimum, as long as a solution actually found is not very much worse. Of course, also other nondeterministic global search methods have been applied to the force-field fitting task, for example simulated annealing[24–26] or artificial neural networks.[27] A complete overview is impossible here.

Hence, it is no surprise that GAs have been frequently used for force-field fitting, since many years. Similar GA approaches also are in use in many other areas, for example in determining the cluster expansion[28,29] for predicting properties of alloys. Not surprisingly, these GA applications share common characteristics. Almost all of the GA applications to force-field fitting, however, were directed at nonreactive force fields, reflecting their prevalence. We are not attempting to give a

[a] H. R. Larsson, B. Hartke
Institut für Physikalische Chemie, Christian-Albrechts-University, Olshausenstr. 40, 24098, Kiel, Germany
E-mail: hartke@pctc.uni-kiel.de
[b] A. C. T. van Duin
Department of Mechanical and Nuclear Engineering, The Pennsylvania State University, University Park, Pennsylvania, 16802, USA

comprehensive overview here but restrict ourselves to a few characteristic examples.

The study that claims to be the first one of this kind was presented in 1998 by Hunger et al.[30] There, GA optimization was used to generate new MM2 parameters, specifically for tripod-Mo$(CO)_3$ compounds, with experimental structural data as reference. To save time in evaluating the force-field data, in an extension of this work[31] a neural network was trained on the results of several previous GA generations, thus saving several orders of magnitude in computational time.

Subsequently, several studies employed GA methods to produce new or better parameters for various chemical species in standard force fields like MM,[32,33] MM3,[34,35] and AMBER[36] as well as in simpler special-purpose force fields.[37,38] In the latter category, there are also early works by one of the present authors,[39–41] which partially predate the publications by Hunger et al. mentioned above. In most of these cases, density-functional theory (DFT) data were used as reference, but obviously also any other level of theory can be used, including CCSD(T).[42] It was also investigated[43] how to adjust technical details of the GA itself (mutation, crossover, population size, and selection) for best performance in such tasks.

In all these GA applications, the immediate difficulty arises that the reference dataset contains a broad variety of items of different nature and with different impact on the parameter optimization. Usually, each item receives a specific weight, and these weights are then included into a sum of squared deviations between force field and reference items, as a single objective, which in turn is minimized. Obviously, there is no unique solution to this task, in the sense that different weights will produce different best solutions (even if there were only one single best solution for each set of weights, and assuming that this solution can be found with certainty).

This problem arises for very many optimizations of practical interest, and multiobjective optimization is frequently advertized as a solution—also for the task of force-field fitting. For example, Handley and Deeth[44] used multiobjective-EA-optimization of a special ligand-field molecular mechanics (LFMM) force field for spin-crossover iron-(II)-amine complexes. Indeed, such methods are able to simultaneously deliver a balanced set of solutions (the Pareto front or set) in which each member excels in one particular sub-objective at the cost of lesser performance in all others. In the end, however, it is still up to the user to decide which of these solutions to actually adopt as the single best one for a specific purpose. Hence, it is not immediately clear that multiobjective approaches are superior to executing single-objective methods several times with different sets of weights.

In all applications mentioned so far, a fixed functional form is pre-assumed for the force field. It typically is not able to adopt all shapes that are mathematically or even physically possible, that is, this choice constitutes a bias and puts a lower bound on the force-field deviation that is greater than zero. Theoretically, it is indeed possible to search for globally optimal functional forms for interatomic potentials in a less biased fashion, while simultaneously fitting parameter values. This was realized[45,46] using a combination of genetic programming, Monte-Carlo sampling, and parallel tempering. These attempts, however, while impressive, have not yet advanced beyond the proof-of-principle stage.

Compared to the huge amount of work on GA optimization of nonreactive force fields, parts of which were just described, almost no work of this kind has been done so far on reactive force fields. Only recently, Angibaud et al.[47] performed a GA optimization of 10 silicon–silicon interaction parameters in a charge-transfer reactive force field for silica,[48] without local search. A later extension to silicon–carbon and carbon–carbon parameters[49] also employed GA optimization. The only other study of this kind[50] that has come to our attention, and the only one that has targeted ReaxFF, will be discussed in some detail below.

This state of affairs is somewhat surprising, considering that already in 1994 (again predating the work by Hunger et al.[30]) Rossi and Truhlar[51] employed a GA to generate new semiempirical neglect of diatomic differential overlap (NDDO) parameters, specifically for the reaction $Cl + CH_4$, in support of the idea of specific reaction parameters (SRP).[52] Clearly, from the viewpoints of parameter optimization and of the later use of the fitted function, there is little difference between Rossi and Truhlar's work on the one hand and reactive force-field fitting on the other hand. In fact, the latter is even simpler, for several reasons: function evaluation is faster and the influence of the parameters on the final energy is more direct. Therefore, the present article helps to fill an obvious gap, by employing GA methods for parameter optimization in a reactive force field of the ReaxFF type. Clearly, employing a global optimization algorithm is even more important for a reactive force field than for a nonreactive one, as the former kind typically has more parameters per atom than the latter.

The more specific purpose of the present contribution is twofold. One aim is to demonstrate the practical feasibility of GA optimization of ReaxFF parameters, for a given test case. The given items consist of the reference dataset, a set of ReaxFF parameters to be optimized, and allowed ranges for all of these parameters. We have investigated how to "tune" the GA for this purpose. It has been criticized[53] that GAs need too much adjustment, compared to other nondeterministic global optimization algorithms, implying that the latter are more robust and hence simpler to use. Our past experience as well as our present findings do not corroborate this. As shown in the section entitled GA Tuning, the influence of GA tuning on GA performance is much smaller than anticipated. Proper adjustment of other aspects of the optimization is far more important, and these adjustments would be needed in exactly the same way for other global optimization algorithms.

A second aim is to explore if previously obtained ReaxFF fitting results can be reached or even improved upon. Most of the presently available ReaxFF parameter sets were obtained using a successive one-parameter parabolic extrapolation technique,[54] here abbreviated as SOPPE (which is not strictly a local search algorithm but definitely also does not pretend to do global search), in conjunction with multiple restarts and a large amount of human experience. With the present contribution, we indeed improve upon previous ReaxFF parameter

sets, hence demonstrating that at least parts of this human fitting experience can be substituted by suitable series of GA runs. This lessens the burden of force-field fitting and makes it more accessible to the nonexpert.

In real-life applications, much of what is prior information here has to be established first: reference data, parameters to be optimized, and their ranges. In preliminary studies,[55] we have already shown that GA parameter fitting also is applicable, efficient, and successful in situations in which this information is missing, for the example of azobenzene in its electronic ground state. Ongoing work is also establishing this for the electronically excited states of the same molecule. In these applications, the techniques established here for SiOH are reused largely unchanged, after obtaining reference data, parameters to be optimized, and their ranges in a simple trial-and-error fashion.

In the final phase of our work for the present project, Pahari and Chaturvedi[50] published a paper in which they also proposed the idea of using GA optimization to fit ReaxFF parameters to reference data. In their study, however, the actual GA optimization is applied only at the very end. The focus is almost exclusively on preparatory steps prior to the GA, namely on determining the smallest set of ReaxFF parameters that have the largest influence on the force-field error, using sensitivity tests and cross-correlation information. Additionally, for the whole study, they have restricted the reference dataset to a single molecule (nitromethane) and four of its decomposition products, for which exclusively geometry data of locally optimized structures are employed as references. Furthermore, they initiate their GA series from a single, predetermined trial vector, and do not employ local optimization. In contrast, in our studies presented here, the focus is on the GA part. We start our initial set of GA calculations with randomized parameter vectors, and we employ the option to hybridize the GA with local search, which turns out to be crucial in the "endgame". Instead of attempting to narrow down the initial set of parameters to be optimized, we stick to it or even attempt enlargements. Last but not least, we employ a large and diverse set of reference data, containing many different species and also very different properties (in addition to some geometry data, also partial charges, crystal cell parameters, and many relative energies of different species). Nevertheless, in real-life situations discussed in the previous paragraph, and if it is possible to quickly calculate the force-field error of very many trial vectors, the techniques presented by Pahari and Chaturvedi definitely are very important and useful.

The remainder of this article is organized as follows. In the section General Methods and Computational Techniques, we provide details for all general approaches employed in this work. The section Optimizing GA Performance focuses on how to "tune" the GA to the optimization case at hand. The actual application of our techniques to the SiOH case is described in the section Production Procedure and Results for SiOH, which is split in a first subsection Production Procedure describing procedural details, and a second subsection Results where we show and discuss the results. The final section provides a summary and an outlook to future work.

General Methods and Computational Techniques

Algorithmic details

In the program design phase for the present project, we anticipated that reference-data-level parallelization would turn out to be a decisive asset to make large-scale global optimization feasible within acceptable real-time. Another design goal was to stay as close as possible to the original ReaxFF implementation. These two goals could only be met by a tight, deep-level interfacing of a GA code and of the ReaxFF code. Therefore, we decided not to use the universal GA program suite OGOLEM,[56] the development of which was started in the Hartke group. Instead, we wrote a new GA implementation which directly addressed suitable pieces of almost unmodified ReaxFF code.

The GA part of the code is based on a standard GA, as described by Goldberg,[23] with the following exceptions: there is no binary encoding of parameter values; instead, the genetic string simply is an array of reals, for all ReaxFF parameter values to be optimized. A single-point crossover is employed; the crossover point is restricted to the parameter boundaries and is determined via evenly weighted random numbers. Mutation of single parameters is realized via normal-distributed deviations from the parent parameter value. The number of parameters to be mutated is an additional external tunable, the mutation amount. Mutation (to a given amount) is always done if crossover does not happen (otherwise, mating would degenerate into a simple but unproductive parent copy). Most importantly, the concept of generations is abandoned; instead, a steady-state or pool model is used.[56,57] From the population of trial solutions, pairs of individuals are extracted quasi-continuously (or, in one of the parallelization options discussed below, in an embarrassingly parallel fashion), selected by suitably designed fitness functions. GA operations are performed upon the selected individuals, and the resulting children are staged for re-entering into the population according to their fitness. If a child is fitter than the currently worst individual in the pool, it replaces the latter; otherwise the child is discarded. To maintain a controllable level of diversity in the pool, diversity measures are introduced via two additional external tunables. One of them is a percentage threshold value that determines if two given values for the same parameter are taken as identical or as different. The second one determines the fraction of all parameters that have to differ (according to the previous measure) such that two individuals are regarded as different. During replacement of children into the pool, it is strictly avoided to arrive at two or more pool individuals that are "identical" according to these criteria. If two "identical" individuals occur, the better one survives, the other one is discarded. After a number of global optimization steps input by the user, each new child with a fitness below a user-supplied threshold value is subjected to a local optimization (discussed in the next paragraph). Following established GA usage in the field of global cluster structure optimization,[58,59] we first started our GA production runs with locally optimizing each and every new child, from the very beginning. However, in

contrast to cluster structure optimization, analytical first derivatives are absent, which renders local optimization expensive. In the present GA application, this increased expense outweighs the increased optimization power. Hence, we changed our approach to performing the first 98–99% global optimization steps without local optimization in parameter space (as "single points"), switching on local optimization only in the final 1–2% of the steps. At the switching point, the whole pool is locally optimized, otherwise selection would render the unoptimized part of the pool obsolete.

Local optimization

As calculation of ReaxFF energies involves an iterative scheme to determine partial charges, and as items in the training set can additionally involve an iterative geometry optimization, there are no simple derivatives of ReaxFF energies with respect to force-field parameters. Therefore, we have tested and employed four different local optimization algorithms without derivative information: an older constraint-free algorithm by Powell, as published in the "Numerical Recipes"[60]; NEWUOA, a constraint-free algorithm by Powell[61]; BOBYQA,[62] a newer variant of NEWUOA with constraints; and the original SOPPE strategy[54] of ReaxFF (which also obeys constraints on the parameter variations). Generally, BOBYQA and NEWUOA were found to be vastly superior to the older Powell algorithm. In fact, also in other applications tested in the Hartke group, BOBYQA displayed superior efficiency. It also outperforms SOPPE; nevertheless, the latter was also employed in the final phases, as due to its rather different iteration directions it may still make progress when BOBYQA stagnates. For practical usage of both BOBYQA and NEWUOA, it is recommended that the ranges of all coordinates of the search space should have similar extent. This is decidedly not the case for the ReaxFF parameters to be optimized here. Hence, for use inside these local optimization routines, all ReaxFF parameters were temporarily rescaled to the interval [0,1].

Parallelization

There are at least three obvious levels of parallelization possible in our setting: (1) across GA individuals, (2) across reference dataset items, and (3) within ReaxFF energy evaluations. The latter is the standard option of calculating interparticle contributions only if they are within distance cutoffs, which breaks the calculation down into quasi-independent, neighborhood-based chunks. This is utilized in the ReaxFF implementation within amsterdam density functional (ADF)[63] as well as in many other force-field-based molecular dynamics (MD) program packages, and will not be discussed further here. The first two options both offer embarrassing parallelism and hence excellent scaling. Option (1) is conceptually trivial within the pool variant of the GA and leads to complete elimination of serial bottlenecks, as the Hartke group has shown earlier.[57] To simplify coding for the present project, this option was not employed here. Instead, we have focused on option (2), as described in the next paragraph.

Using explicit messaging via the message-passing interface (MPI), we have set up a straightforward master–slave model. For every given GA individual (which simply is a single set of ReaxFF parameter values), the master hands out reference data items to the slaves which produce the corresponding ReaxFF comparison values for them. The advantage of the master–slave paradigm is that only the master steps through all GA-related code parts (which are not compute-intensive, compared to the ReaxFF calculations), while the slaves enter the reference item treatment section once at the beginning of program execution and then stay trapped there until the end of the program. This ensures a clean program code, despite the very basic MPI parallelization. Handing out reference data items is realized via a simple loop that ensures automatic load balancing[64] In our initial implementation, only single reference data items are handed out. As the computational time needed for many of them is rather short (cf. section Reducing Test Set Size for GA Tuning), increasing the number of slaves beyond a certain number will diminish overall speedup due to communication overload at the master. Nevertheless, we can achieve reasonable speedups with a few dozen slave processes.

Test cases

We have started to perform global GA optimization of ReaxFF parameters for three test cases: Co, SiOH, and Gly. For each of these cases, the van Duin group prescribed a reference dataset, a set of ReaxFF parameters to be optimized, and lower/upper limits within which the ReaxFF parameters were allowed to be varied. With some exceptions described in the section Fitting Procedure for Parameter Ranges and the section Production Procedure, we have not changed these presettings.

In all of these cases, we have used the same general measure for deviations between the force-field data and the reference data, which we will abbreviate *error sum* in the following:

$$\text{error sum} = \sum_{i=1}^{n} \left(\frac{x_{i,\text{calc}} - x_{i,\text{ref}}}{\sigma_i} \right)^2 . \quad (1)$$

$x_{i,\text{ref}}$ is the value of a reference data item, and $x_{i,\text{calc}}$ is the corresponding value as calculated from ReaxFF. σ_i serves several purposes. Formally, it is a standard deviation and a way to make each summation term dimensionless. Practically more important is that the user is free to introduce arbitrary numerical values for it, which makes it possible to tune the relative weight of each item as needed. Therefore, individual error sum values are rather meaningless, they could be scaled to any number larger than zero. Additionally, due to the different number and nature of reference items, they are not comparable anyway between different cases. Hence, their only purpose is to gauge relative optimization progress within one given case.

The Co test case is based upon recent work in the van Duin group,[65] where a ReaxFF parametrization for cobalt was developed and shown to yield good descriptions of the energetics and properties of various crystal phases, amorphous configurations, clusters, vacancies, and surfaces. Twelve ReaxFF parameters were to be varied, and the reference dataset

consisted of 147 items. Without using further information, and with minimal computational effort, the best error sum of 1443.7 achieved previously by the van Duin group could be improved in our GA optimizations to 1439.8; no further progress seems to be possible. Hence, compared to the SiOH case treated in the remainder of this article, parameter optimization tasks with a complexity of this Co case pose no challenge to our approach.

The Gly test case is work in progress. Its details and current status will be mentioned briefly in the conclusion section.

The remainder of this article is focused on the SiOH test case, which poses a surmountable challenge to our approach. This test case is based upon earlier work of the van Duin group[66,67] where a ReaxFF parametrization for materials involving silicon and silicon oxides was developed. In its original form, it contains 67 parameters to be optimized, 304 reference structures (ranging from simple molecules like silanes to periodic solid-state structures like quartz), and a set of 309 reference properties for them (the training set, consisting of geometry data, partial charges, crystal cell parameters, and relative energies).

Factors influencing GA performance

In real-life applications of the present global parameter optimization scheme, there is less information available in the beginning. Namely, it is as yet (largely) unknown which reference data items should be used, which ReaxFF parameters should be varied, and within which limits these variations should happen. It will be a recurring theme of the next two sections that these items have a decisive influence not only on the final force-field quality but also on the overall efficiency of the GA optimization. To take the example of parameter variation ranges, it is clear that varying the ranges directly influences the size of the search space volume to be covered by the optimization. Trivially, a search in a smaller space is much more efficient than a search in a larger one. Therefore, starting from good guesses for the parameter ranges is vital for obtaining good results within short real times. Conversely, allowing the full range $[-\infty, +\infty]$ for all parameters would be a bad idea (disregarding the possibility that this may lead to unphysical characteristics of various force-field terms, for many parameter value realizations). Fortunately, it turns out that a straightforward iteration of the whole GA scheme allows for narrowing down the variation ranges even in the absence of prior information on them. This range-fitting procedure will be explained in the section Fitting Procedure for Parameter Ranges and applied in the section Production Procedure and Results for SiOH.

As already mentioned in the introductory section, another decisive lever influencing the global optimization is to change the number of parameters to be optimized, or their selection. How to do this systematically and successfully has already been demonstrated by other authors.[50] In the present study, we employed this option to generate smaller test scenarios, to make GA tuning faster (cf. section Reducing Test Set Size for GA Tuning), and to attempt a significant enlargement of the set of parameters to be optimized, to further improve the error sum (cf. section Production Procedure). The central idea of this study, however, was to take the set of parameters to be optimized as given and immutable, even if it presents us with a large search space, and to demonstrate feasibility of GA optimization within this given search space.

Optimizing GA Performance

Reducing test set size for GA tuning

As quantitative measurement of the quality of the force field, the error sum of squares was used, defined as already shown above in eq. (1). The values of the reference data items, $x_{i,\mathrm{ref}}$, were obtained from B3LYP/6–31G** calculations.[66] Various data items are possible, for example bond lengths, partial charges, or relative energies, all these either for fixed input geometries or evaluated after a geometry optimization. The comparison values $x_{i,\mathrm{calc}}$ are calculated in the same fashion (i.e., possibly preceded by a geometry optimization) from ReaxFF.

To obtain good performance of the GA, its parameters (some of which were mentioned in section General Methods and Computational Techniques) needed to be optimized. The SiOH training set consists of 304 individual computation jobs for comparison with the reference data. This turned out to be too much for a quick but thorough investigation of the behavior of the GA parameters and needed to be reduced for this purpose. Hence, the time needed for each of the 304 computations for 247 randomly created force-field parameter vectors was retrieved and compared. Most of the jobs needed only milliseconds but some of them, for example, geometry optimizations of periodic structures, up to 2 min. Removal of all jobs that needed more than 0.5 s (90 jobs altogether) yielded a time saving of about 95%. However, the test cases for the GA parameter optimization should be small, that is, not time consuming, but also representative. Therefore, an additional test case was obtained by removing less jobs and by ensuring that the reduced training set still consisted of jobs of different types. Forty-three jobs were removed which still yielded a useful time saving of 87%.

To get insights into the behavior of the GA by changing the dimension of the optimization problem, the set of 67 to be optimized parameters was reduced to 40. With these reductions, four test cases were obtained, each of them with 214 or 261 jobs in the training set and 40 or 67 parameters to be optimized. Table 1 sums this up.

GA tuning

As in all indeterministic global optimization algorithms of this kind, there is no true convergence criterion. Further progress does diminish progressively, but ultimately the user has to decide when to stop, and this depends on the problem at hand. After some tests containing 20 different GA runs with a randomly created starting population and a maximum iteration number of 1000 with each of the four test cases and

Table 1. Scheme of the test settings.

Case	Parameters to be optimized	Training-set-jobs
bo	67	261
bp	40	261
so	67	214
sp	40	214
Original	67	304

three runs with a maximum iteration number of 10,000 with the case "so" (see Table 1), a total iteration number of 800 was found to be sufficient to reach the region where progress had essentially leveled off.

Then, 19 tests were performed, with a systematic change of one GA parameter. Each of these tests consisted of 10 GA runs for each of the four test cases, resulting in a total of 760 single GA test runs. Afterward, 17 additional tests were performed with a systematic change of two parameters at once, to check for correlations between parameters. The analyzed parameters were the initial pool size, the overall pool size (which can be different from the initial pool size, that is, the pool is initially allowed to grow), the crossover rate, the mutation amount, and the two diversity threshold parameters mentioned in the section Algorithmic Details.

To give an example, Figures 1 and 2 show the results for a semirandom selection of some test runs of setting "bo" with varying crossover probability and mutation amount. In these kinds of figures, we plot the error sum of the currently best individual versus the global iteration counter. The curves exhibit a staircase-like shape as the best individual is not

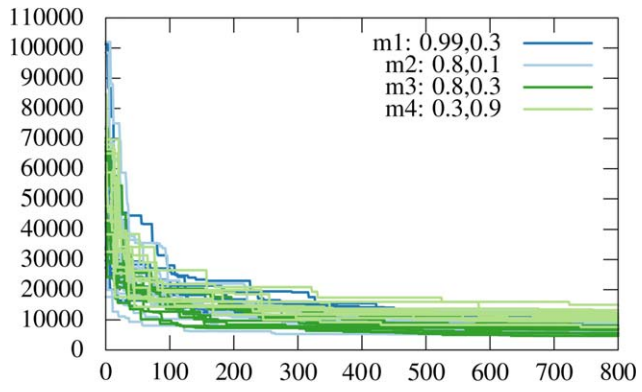

Figure 1. Selection of test runs with varying crossover probability and mutation amount. For each iteration in 10 runs of each kind, the error sum of the currently best individual is shown. The internal identifier is given for each run, as well as the crossover probability and the mutation amount. The color scheme for this figure and for later ones was taken from [68].

improved at every step but is guaranteed to survive if it is not improved upon, via the automatic elitism inherent in the pool strategy.

Surprisingly, all runs shown performed similarly, although the crossover probability and the mutation amount changed drastically for these runs: from 0.99 to 0.3 and from 0.3 to 0.9, respectively. The same finding was obtained from our other

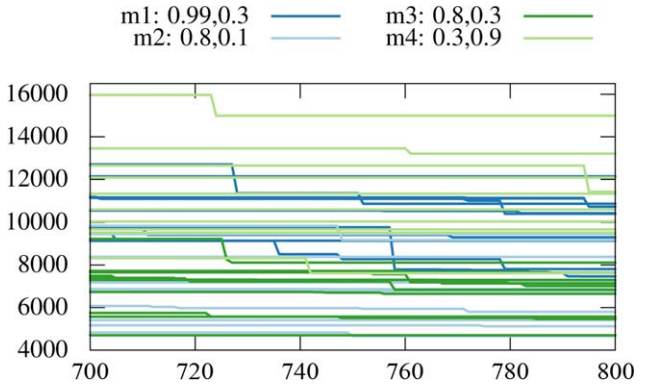

Figure 2. Enlarged final portion of Figure 1, for global iterations 700 through 800, to make the individual traces better visible.

GA parameter test runs. These results are counter-intuitive if one is familiar with the literature where the values of the GA parameters and the fine details of GA operator design are discussed extensively.[43,69–72] While the change in performance by varying each GA parameter was smaller than expected, it was still there. Upon first glance at Figure 1, one clearly sees that case "m4" (excessive mutation) performed slightly worse than the others, that is, there was less decrease in the error sum of the best individual during the iterations than in other runs. Likewise, it is discernible that case "m1" (excessive crossover) also performed worse than the remaining two cases, especially in the beginning, until about 500 iterations. The limited performance of these two cases is a sign that both mutation and crossover indeed worked as intended, that is, they were not superfluous decoration.

It is quite hard to decide whether case "m2" or "m3" were better. Upon closer inspection of Figure 2, one sees that some runs of case "m2" arrived at better results than the runs of case "m3". However, there were also runs that performed worse and ended up at error sums between 10,000 and 8000. It is not clear whether these findings should be attributed to the limited number of runs for each case (10) or indeed were an outcome of the changed mutation amount.

Investigations like these were performed for all 36 $\times$ 4 tests mentioned above. Analysis of the results was assisted by an exponential fitting of all 10 runs for each test. Overall, the following parameter combinations showed the best performance for all four test settings mentioned in Table 1: a population size of about 30; a crossover probability of 0.8; a mutation amount of 0.3; a value of 0.01 for the threshold for taking scaled parameters as identical and 80% as the amount of parameters that has to differ for two individuals to be counted as different (see section Algorithmic Details for further explanations).

The choice of population size was not as clear as the crossover/mutation choice illustrated above. Runs with population sizes of 27 and 100 gave similar results for the best individuals. For a larger population, the population-filling in the beginning of an optimization (i.e., creating individuals and calculating their error sum) needs more time but then the diversity is larger. During the production phase for the results shown in section Production Procedure, it became clear that the

convergence was worse for large numbers of iterations and a predetermined starting population, and it was shown that the improved diversity with this population size was not really needed.

As it was shown above, it makes no sense to discuss whether, for example, the crossover probability should be 0.8 or 0.85, as the difference in performance is normally negligible. However, the parameter set quoted two paragraphs earlier (30, 0.8, 0.3, 0.01, 80%) as the best one of our GA tuning proved itself during production runs with the nonreduced training set mentioned in section Production Procedure. It did show better performance than an *ad hoc* configured setting with the values 27, 0.7, 0.2, 0.1, and 80%. It should be mentioned that an adjustment, for example, of the diversity parameters can be worthwhile during runs in later production phases with a prearranged starting population with good error sums, to avoid premature convergence.

Furthermore, four different modes of selecting pairs of individuals for mating from the error-sum-ordered pool were tested, using the case "bo". One selection mode was to have two exponential fitness functions with different slope, reflecting the combined need for exploitation (steep function strongly favoring the best) and for exploration (less steep function also giving worse individuals a chance). Another ansatz was to use step functions.[69,73] Hence, the second selection mode was to use a Heaviside function for the first individual, with an equal probability for the first half of the population, and to select the second individual with an equal probability across the whole population. The third selection mode used the mentioned Heaviside function for both selections. The fourth and last mode employed an eight-step function for both selections. The individuals in the first eighth of the ordered population had a probability of 8/36, the individuals in the second eighth had a probability of 7/36 and so on. Surprisingly, all four ansätze showed similar behavior, despite their varieties. This was unexpected because the selection function is said to have a decisive influence on the behavior of the global optimization.[69–73] Within the small performance differences found, the first selection mode mentioned above produced somewhat better results, followed by the third one.

Fitting procedure for parameter ranges

Another sensitive lever is the variation ranges prescribed for each force-field parameter, as these directly influence the size of the search space. The ranges supplied by the van Duin group were not based on investigations by a GA, hence it was conceivable that the GA could yield better results with a larger search space. Additionally, in real-life applications of our GA parameter optimization, these ranges usually are not available initially and have to be established prior to or during the optimization. Therefore, test runs were started using the originally given ranges with enlargement factors of 4 and 8, respectively, with the test cases "so" and "bo". Not surprisingly, some of these drastic enlargements led to unphysical behavior of the corresponding force-field terms, which in turn induced program crashes. Therefore, the ranges of 14 parameters were

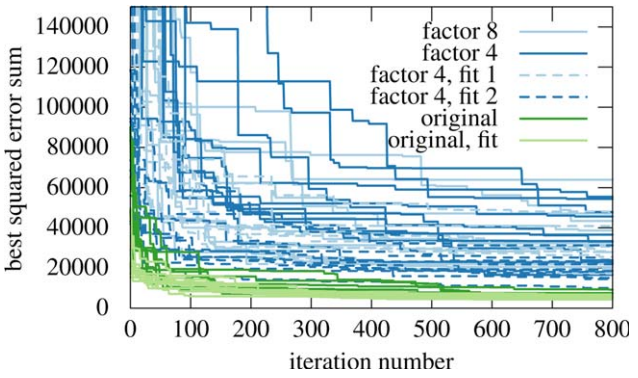

Figure 3. Results of the fitting procedure for parameter ranges, as explained in the text. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

only doubled. From the results of a first round of GA runs with the factor-4-enlarged ranges and from runs of the "original" ranges from the van Duin group, new ranges were obtained by using the minimum and maximum values of each parameter of the best individual obtained from the 10 runs. In the following, this strategy is referred to as "fitting procedure". This procedure was repeated for the factor-4-enlarged ranges. The result can be seen in Figure 3. Obviously, a massive enlargement of each parameter range led to significantly worse performance. For example, after 800 iterations, the runs with an enlargement of factor 8 yielded individuals with error sums similar to the randomly created initial individuals of the runs with the "original" ranges obtained from the van Duin group. However, the fitting procedure of the ranges was quite successful. After two iterations of it, the ranges, which originally were enlarged by factor of 4, led to runs which gave results comparable with the results of runs with the original van Duin ranges. A fitting procedure of the latter runs produced even better results.

Production Procedure and Results for SiOH

Production procedure

According to the results of the section Optimizing GA Performance and preliminary tests prior to that, a rough strategy was evolved. At first, 12 production runs were executed, with 4900 iterations without local optimization, followed by 100 iterations with local optimization using the BOBYQA-routine and a loose convergence criterion. The iteration number was larger than in the procedures in the section GA Tuning, to exploit the most of each run. As noted, the strong initial progress typical for quasi-random search algorithms keeping track of the best-so-far solution leveled off at about 600 to 800 iterations. However, it is also typical that such algorithms still show occasional progress in the later stages, which is worthwhile to attain during production, unless the overall timing becomes excessive. The population sizes were 30 and 100 (see the discussion in section GA Tuning). The parameter ranges were obtained by the "fitting procedure" of the test runs with the original ranges, see the section entitled Fitting Procedure for

Parameter Ranges. Afterward, the best individuals of each run (about 400) were selected for an additional optimization with the NEWUOA-routine and a tighter convergence criterion. As mentioned in section Local Optimization, the (more compute-intensive) NEWUOA-routine has no constraints so that parameters with values near the artificial parameter bounds can "relax" beyond those bounds. Beginning with randomly created individuals with error sums between 2×10^3 and 1×10^7, the optimization ended after this phase at a best error sum of 4268.

Then, a new starting population was selected from these optimized individuals, according to criteria of small error sum and large diversity, that is, large Euclidean distances (L^2 distance) between one individual vector and all other vectors. The ranges of each parameter were fitted to this starting population. New runs were executed with two selected populations with a size of 30 and 122, respectively, and the procedure mentioned above was repeated. Although the starting population was specified and not randomly created in this phase, the optimization runs showed quite unequal error sums for the best individual, which stresses the indeterministic character of global optimization. During these optimizations, we experienced a higher error sum after local optimization with the BOBYQA, the SOPPE, and the older Powell routine (cf. section Local Optimization). This should not happen in a minimization routine but (without explicit tests) it cannot be strictly avoided when finite-size steps are taken, and in particular if no analytical gradient information is available. As we observed such behavior only for vectors with small error sums, we concluded that this is also a symptom of being close to the lower limit of global optimization. Only the NEWUOA-routine showed small optimization success.

Additionally, the runs needed more iterations for a substantial decrease of the error sum—iteration numbers between 15,000 and 30,000 were common. However, also the time needed for each iteration was diminished drastically, because all individuals in the population already had good error sums. Therefore, geometry optimizations that were part of the error sum calculation needed less iterations because the minimum geometry with the ReaxFF-potential differed not so much from the starting-point geometry obtained by quantum-mechanical calculation. An optimization run with the random populations took about seven days, whereas a run with the starting population selected as explained and with 10 times more iterations with local optimization only took about three days. It was also observed that at later stages of these runs, a lot of children with very large error sums were created by crossover or mutation. Therefore, the program was slightly adjusted so that only children with sufficiently small error sum were locally optimized. Additionally, only the best 30% of the population were locally optimized during the initialization of the local optimization phase, to accelerate the local optimization phase.

After three iterations of this procedure with an increasing GA-iteration number and a decreasing convergence criterion for the local optimization routines, an individual with an error sum of 3196 was finally obtained.

As progress to even better results appeared to become increasingly difficult, we decided to test if this error sum could be further improved upon by allowing more parameters than the originally preset 67 to be optimized. Based on the purpose of each force-field parameter, 124 additional parameters were selected that could have an influence on the SiOH force field. Therefore, 191 parameters now needed to be optimized. The first step to achieve this was to set up ranges for the extra parameters. To simulate a realistic situation where knowledge about parameter ranges may be scarce or absent, we resorted to comparing values of these parameters with values of similar parameters or of other force fields, and arranged ranges based on that. However, randomly created individuals with parameter values within these ranges led to force fields with unphysical behavior and program crashes. After further manual adjustment, the ranges mostly yielded "stable" individuals but with very high error sums. Therefore, 30 GA runs with 2000 iterations in each case were executed and the best 20 individuals with error sums at about 3×10^5 were used to fit the parameter ranges for the first time. Then, two production rounds were performed, following the aforementioned procedure, with iterations between 5000 and 10,000. For the optimization runs with 67 parameters to be optimized, all values of other parameters were fixed and therefore the same for all individuals. Hence, only one (the best) vector of the outcome of the optimization process with 67 parameters to be optimized was used and added to the starting population in the second production round. Otherwise, there would have been a change in the values of the 124 extra parameters to be optimized only due to mutation, but not to crossover, which is the main operator of the GA.

During this production, the values of the ranges for each parameter were successively inserted into some individuals from the global optimization, and the error sum was calculated, to check whether the parameter has an influence on the error sum or not. Thirty-four of the additional parameters showed no large influence and were, therefore, excluded from further optimization. Otherwise, these parameters could have arrived at not controllable values within their ranges. This would have had no influence on the behavior of the molecules tested in the training set but it could potentially affect the force-field performance for other molecules outside the training set.

After local optimization, an individual with the smallest error sum of 2807 was obtained. By doing further optimizations, this error sum could have been improved further. However, this procedure was only meant as a demonstration of what can be achieved when focusing on a single system. The individuals that were obtained by an optimization of the extended set of 191 parameters cannot be used outside of the SiOH-force field, as other force fields also use those additional parameters but with different values (cf. section Introduction).

Results

In Figure 4, a more fine-grained view on the results is given, in terms of the individual weighted entries comprising the error

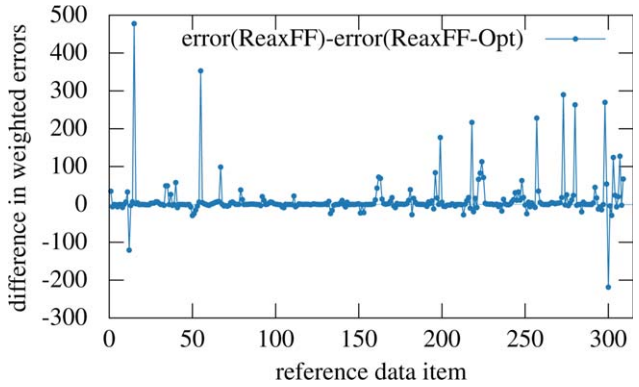

Figure 4. Differences in individual error sum contributions [eq.(1)] between the original SiOH ReaxFF force field [67] ("ReaxFF", error sum of 6455) and our newly GA-fitted ReaxFF force field ("ReaxFF-Opt", error sum of 2807), as a function of the (arbitrary) ordinal number of the entries in the training set. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

sum of eq. (1), shown as the difference between the error items of the originally published SiOH-force field[67] and the corresponding ones of our newly GA-fitted one. Most of these error differences are close to zero, that is, our new results do not constitute a dramatic improvement. This is to be expected; it re-emphasizes that the production of the original SiOH-force field was done with care and already provides a very good fit to the reference data, leaving little room for further improvement, within the possibilities of the ReaxFF functional form. Nevertheless, there are about two dozen reference data items where there are substantial differences in the weighted errors. For only two of them (no. 12 and 300), the graph in Figure 4 peaks into the negative direction, meaning that the error in our new fit is larger than it was before. In all other cases (positive peaks), the error contributions have become smaller. In addition to the overall error sum values, this is further support for our claim that this new fit actually is better. To properly judge this outcome, it should be recalled that the originally published force field already was the well-converged final result of a diligent, compute-intensive project, performed by experienced researchers in the group of the ReaxFF inventor.

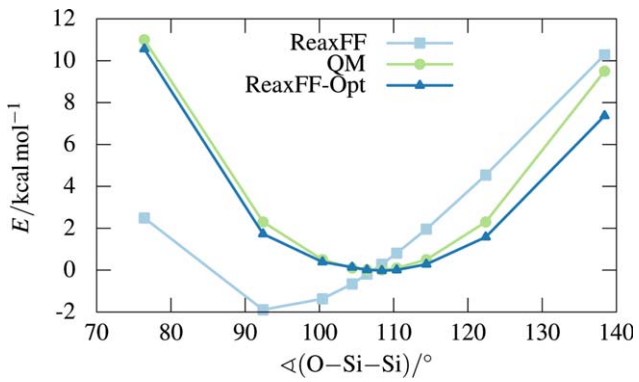

Figure 5. Potential energy as a function of O—Si—Si bond angle in HO—SiH_2—SiH_3, for the reference data (B3LYP/6–31G**) (green), the original SiOH ReaxFF data [67] (light blue, error sum of 6455), and the newly GA-fitted ReaxFF data (dark blue, error sum of 2807). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

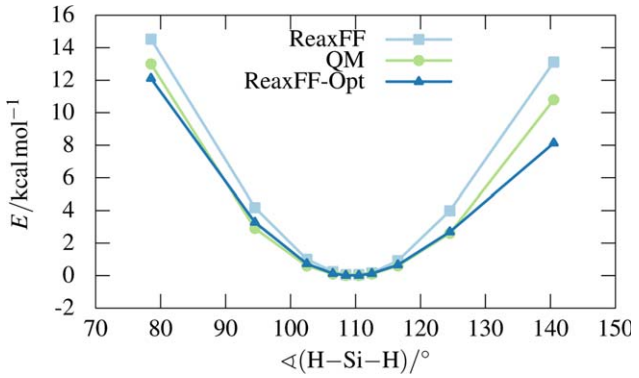

Figure 6. As Figure 5, for the H—Si—H bond angle in SiH_4. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

The GA fitting in the Hartke group, however, started with no experience in ReaxFF force-field fitting, and only took about two months real-time within a BSc thesis project,[74] including all the strategy-exploration work described partially in the section Optimizing GA Performance. If the calculations necessary for the complete GA fitting were redone by us now, they would take about 2 weeks on 200 processors.

Further insights into the new GA-fitted results become accessible in still more fine-grained data. To this end, we show a semirandom selection of potential energy surface cuts along certain coordinates, for certain molecular systems from the training set. Obviously, in some cases, our GA fit leads to substantial improvements, even of a qualitative nature. In others, only smaller quantitative improvements are visible or even no improvement at all, despite qualitative defects compared to the reference data.

We show four examples of this kind in Figures 5–8. In Figure 5, our newly optimized ReaxFF data obviously constitute a substantial improvement over the previously published ones,[67] not just quantitatively but even qualitatively. In Figure 6, the diagnosis is less clear, but upon closer inspection only the rightmost data point prevents us from reaching the same conclusion again. In Figure 7, both ReaxFF curves manage to get the location of the minimum correctly, but fail qualitatively

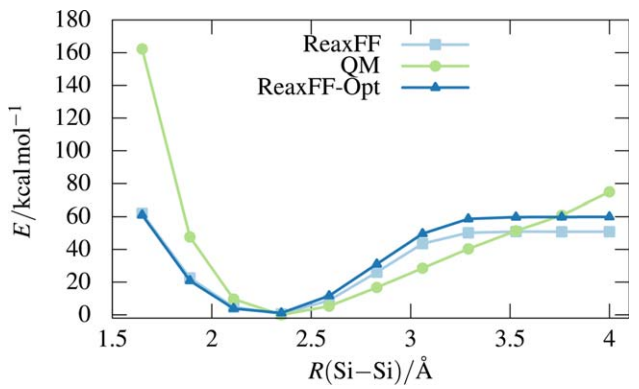

Figure 7. As Figure 5, for the Si—Si bond distance in H_3Si—SiH_3. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

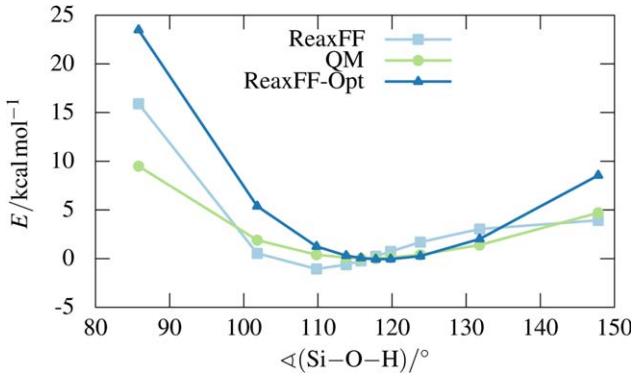

Figure 8. As Figure 5, for the Si—O—H bond angle in H_3Si—O—H. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

in other regions; our new fit is no improvement here. In Figure 8, the overall shape of our new ReaxFF curve is better but the data points of the old force field stayed closer to the reference data overall.

For interested readers, we provide our two best force fields (with error sums of 2807 and 3196, respectively) and full information on the training set (geometries and properties) and on the parameters to be optimized (including their variation ranges) in the Supporting Information.

Conclusions

Summary

We have presented an unbiased, global GA optimization of a ReaxFF force field for SiOH species. A set of 304 reference structures (containing a mixture of species, ranging from simple silanes to quartz) and a set of 309 reference properties for them (a mixture of geometry data, partial charges, crystal cell parameters, and relative energies) were taken as given, as well as a realistic set of 67 force-field parameters to be optimized, within given ranges.

Beyond these data, no further prior information was utilized in the GA search. Nevertheless, our best GA-optimized force fields constitute further improvements upon the (already good) SiOH-ReaxFF force fields published previously. For the force field from Ref. [67], the error sum was 6455, but later reoptimizations using SOPPE, prompted by our GA work, could bring this number down to just under 4000. Our best GA results range between 2800 and 3200. From the behavior observed in our optimizations, we assume that no significant further improvement will be possible, with the selection of parameters and reference data items set by the van Duin group. Error sum values of 2800 (and possibly lower) are accessible with an extension of the set of parameters to be optimized, to a total of 191, but this compromises the compatibility of the resulting SiOH force field with other ReaxFF parametrizations, and hence was not pursued further.

As no expert knowledge about the force field in general or the purpose of its parameters in particular is necessary for our GA parameter optimization approach, it provides easier access to force-field optimization for less experienced users, and it also helps experienced force-field optimizers to arrive at better solutions, closer to global minima. As one step toward end-user distribution of our code, integration of it into the ADF package[63] currently is underway.

In GA applications of this kind, detailed tuning of the GA is less important than providing suitable ranges in which the force-field parameters are allowed to vary. Nevertheless, as demonstrated here, it is also possible and realistic to iteratively improve upon badly guessed initial parameter ranges: A first round of GA runs is started with the initially guessed variation ranges (which can be considerably too wide) and with random initialization of the population within these ranges. From the resulting final populations, the best individuals are selected. Their parameter value variations allow for a realistic estimate of narrower variation ranges. These best individuals constitute the starting population for a second round of GA runs, employing the narrower variation ranges, which improves GA efficiency considerably. After a few iterations, the variation ranges have shrunk to intervals that allow for arriving at results of the same quality as with more suitable initial parameter ranges.

Outlook

The Gly test case mentioned briefly in the section Test Cases is also based upon recent work in the van Duin group,[75] where a ReaxFF parametrization for the amino acid glycine was developed, in order to investigate the water-mediated neutral ⇌zwitterionic tautomerization in aqueous solution. Here, the reference dataset consists of 2049 items, and 299 parameters are to be varied, producing a very much larger search space than for the other two cases. Work on the Gly test case is in progress. From its current status, we cannot yet decide whether the same techniques demonstrated here for the SiOH test case also allow us to finish the Gly test case successfully or if such a large search space would need impracticably large amounts of computer time. One possible remedy in the latter situation may be iterative cycling over smaller-sized subproblems, which was shown to be successful in cluster structure optimization.[76]

As noted in the section Parallelization, for our current master–slave program version, parallel speedup is worthwhile for a few dozen slave processes. In future developments of our code, we plan to change this by handing out chunks of several reference data items. This will improve the ratio of computation to communication and hence will allow for better scalability. This will be improved upon further by also implementing the (simpler) parallelism at the level of GA individuals. Parallelism at the lowest level, at the calculation of ReaxFF energies, will be included only later, as it is most worthwhile only for large systems, which do not tend to be part of reference datasets for parameter fitting.

For each individual and independent of parallelization issues, further speedup can be realized by calculating the error sum "on the fly", as each reference data item is returned to the master process. Currently, this is done only after every

item has been processed. The on-the-fly option would make it possible to refrain from further (probably costly) evaluations of reference data items if the error sum already is too large to be competitive. Of course, this destroys some information on the shape of the cost function, but only in its worst regions. It can be argued that the guidance offered by these regions is limited to nonexistent anyway, in a scheme like a GA where the best schemata tend to be strongly favored. No information of any consequence is lost, however, if the on-the-fly error sum cutoff is placed at the bad end of the ordered pool. In test calculations for the SiOH case, performed after the studies described in this article, this has incurred substantial savings in computer time. It remains to be seen if this can be generalized.

Acknowledgment

BH thanks Alexei Yakovlev (SCM Company, The Netherlands) for his patient and competent help with incorporating our techniques into the ADF program package. ACTvD acknowledges funding from the US National Science Foundation/SI2 grant #1047857.

Keywords: global optimization · force-field fitting · genetic algorithms · evolutionary algorithms · reactive force fields

How to cite this article: H. R. Larsson, A. C. T. Duin, B. Hartke. *J. Comput. Chem.* **2013**, *34*, 2178–2189. DOI: 10.1002/jcc.23382

[1] P. Vashishta, R. K. Kalia, A. Nakano, *J. Nanopart. Res.* **2003**, *5*, 119.
[2] D. E. Shaw, P. Maragakis, K. Lindorff-Larsen, S. Piana, R. O. Dror, M. P. Eastwood, J. A. Bank, J. M. Jumper, J. K. Salmon, Y. Shan, W. Wriggers, *Science* **2010**, *330*, 341.
[3] T. J. Lane, D. Shukla, K. A. Beauchamp, V. S. Pande, *Curr. Opin. Struct. Biol.* **2013**, *23*, 58.
[4] D. W. Brenner, *Phys. Rev. B* **1990**, *42*, 9458.
[5] Y. T. Chang, W. H. Miller, *J. Phys. Chem.* **1990**, *94*, 5884.
[6] J. Aqvist, A. Warshel, *Chem. Rev.* **1993**, *93*, 2523.
[7] Y. Kim, J. C. Corchado, J. Villa, J. Xing, D. G. Truhlar, *J. Chem. Phys.* **2000**, *112*, 2718.
[8] S. C. L. Kamerlin, J. Cao, E. Rosta, A. Warshel, *J. Phys. Chem. B* **2009**, *113*, 10905.
[9] S. C. L. Kamerlin, A. Warshel, *Faraday Discuss.* **2010**, *145*, 71.
[10] H. S. Johnston, *Adv. Chem. Phys.* **1961**, *3*, 131.
[11] B. C. Bolding, H. C. Andersen, *Phys. Rev. B* **1990**, *41*, 10568.
[12] L. Ojamae, I. Shavitt, S. J. Singer, *J. Phys. Chem.* **1998**, *109*, 5547.
[13] P. Ballone, R. O. Jones, *J. Chem. Phys.* **2004**, *121*, 8147.
[14] A. C. T. van Duin, S. Dasgupta, F. Lorant, W. A. Goddard, III., *J. Phys. Chem. A* **2001**, *105*, 9396.
[15] S. S. Han, J. K. Kang, H. M. Lee, A. C. T. van Duin, W. A. Goddard, III, *J. Chem. Phys. 123*, 2005, 114703.
[16] K. D. Nielson, A. C. T. van Duin, J. Oxgaard, W. Q. Deng, W. A. Goddard, III, *J. Phys. Chem. A* **2005**, *109*, 493.
[17] K. Chenoweth, A. C. T. van Duin, P. Persson, M. J. Cheng, J. Oxgaard, W. A. Goddard, III, *J. Phys. Chem. C* **2008**, *112*, 14645.
[18] J. A. Keith, D. Fantauzzi, T. Jacob, A. C. T. van Duin, *Phys. Rev. B* **2010**, *81*, 235404.
[19] A. Nakano, R. K. Kalia, K. Nomura, A. Sharma, P. Vashishta, F. Shimojo, A. C. T. van Duin, W. A. Goddard, III, R. Biswas, D. Srivastava, *Comput. Mater. Sci.* **2007**, *38*, 642.
[20] A. Neumaier, *Acta Numer.* **2004**, *13*, 271.
[21] A. Neumaier, O. Shcherbina, W. Huyer, T. Vinkó, *Math. Program. Ser. B* **2005**, *103*, 335.
[22] J. H. Holland, Adaption in Natural and Artificial Systems, University of Michigan Press: Ann Arbor, MI, **1975**.
[23] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley: Reading, MA, **1989**.
[24] Y. A. Arnautova, J. Pillardy, C. Czaplewski, H. A. Scheraga, *J. Phys. Chem. B* **2003**, *107*, 712.
[25] E. R. Hatcher, O. Guvench, A. D. MacKerell, Jr., *J. Chem. Theor. Comput.* **2009**, *5*, 1315.
[26] W. Yu, P. E. M. Lopes, B. Roux, A. D. MacKerell, Jr., *J. Chem. Phys.* **2013**, *138*, 034508.
[27] H. M. Marques, I. Cukrowski, *Phys. Chem. Chem. Phys.* **2002**, *4*, 5878.
[28] V. Blum, A. Zunger, *Phys. Rev. B* **2005**, *72*, 020104.
[29] B. Arnold, A. D. Ortiz, G. L. W. Hart, H. Dosch, *Phys. Rev. B* **2010**, *81*, 094116.
[30] J. Hunger, S. Beyreuther, G. Huttner, K. Allinger, U. Radelof, L. Zsolnai, *Eur. J. Inorg. Chem.* **1998**, 693.
[31] J. Hunger, G. Huttner, *J. Comput. Chem.* **1999**, *20*, 455.
[32] T. R. Cundari, W. T. Wi, *Inorg. Chim. Acta* **2000**, *300*, 113.
[33] B. Courcot, A. J. Bridgeman, *J. Comput. Chem.* **2011**, *32*, 240.
[34] T. Strassner, M. Busold, W. A. Herrmann, *J. Comput. Chem.* **2002**, *23*, 282.
[35] M. Tafipolsky, R. Schmid, *J. Phys. Chem. B* **2009**, *113*, 1341.
[36] J. M. Wang, P. A. Kollman, *J. Comput. Chem.* **2001**, *22*, 1219.
[37] A. Globus, M. Menon, D. Srivastava, *Comput. Model. Eng. Sci.* **2002**, *3*, 557.
[38] C. R. Herbers, K. Johnston, N. F. A. van der Vegt, *Phys. Chem. Chem. Phys.* **2011**, *13*, 10577.
[39] B. Hartke, *Chem. Phys. Lett.* **1996**, *258*, 144.
[40] B. Hartke, *Theor. Chem. Acc.* **1998**, *99*, 241.
[41] B. Hartke, M. Schütz, H.-J. Werner, *Chem. Phys.* **1998**, *239*, 561.
[42] J. M. Dieterich, B. Hartke, *J. Comput. Chem.* **2011**, *32*, 1377.
[43] B. C. Barnes, L. D. Gelb, *J. Chem. Theor. Comput.* **2007**, *3*, 1749.
[44] C. M. Handley, R. J. Deeth, *J. Chem. Theor. Comput.* **2012**, *8*, 194.
[45] A. Slepoy, M. D. Peters, A. P. Thompson, *J. Comput. Chem.* **2007**, *28*, 2465.
[46] W. M. Brown, A. P. Thompson, P. A. Schultz, *J. Chem. Phys.* **2010**, *132*, 024108.
[47] L. Angibaud, L. Briquet, P. Philipp, T. Wirtz, J. Kieffer, *Nucl. Instrum. Methods B* **2011**, *269*, 1559.
[48] L. Huang, J. Kieffer, *J. Chem. Phys.* **2003**, *118*, 1487.
[49] L. G. V. Briquet, A. Jana, L. Mether, K. Nordlund, G. Henrion, P. Philipp, T. Wirtz, *J. Phys. Condens. Matter* **2012**, *24*, 395004.
[50] P. Pahari, S. Chaturvedi, *J. Mol. Model.* **2012**, *18*, 1049.
[51] I. Rossi, D. G. Truhlar, *Chem. Phys. Lett.* **1994**, *233*, 231.
[52] A. Gonzalez-Lafont, T. N. Truong, D. G. Truhlar, *J. Phys. Chem.* **1991**, *95*, 4618.
[53] S. E. Schönborn, S. Goedecker, S. Roy, A. R. Oganov, *J. Chem. Phys.* **2009**, *130*, 144108.
[54] A. C. T. van Duin, J. M. A. Baas, B. van de Graaf, *J. Chem. Soc. Faraday Trans.* **1994**, *90*, 2881.
[55] H. R. Larsson, B. Hartke, *Comput. Methods Mater. Sci.* **2013**, *13*, 120.
[56] J. M. Dieterich, B. Hartke, *Mol. Phys.* **2010**, *108*, 279; Available at http://www.ogolem.org/, last accessed: 6 July 2013.
[57] B. Bandow, B. Hartke, *J. Phys. Chem. A* **2006**, *110*, 5809.
[58] B. Hartke, *Angew. Chem. Int. Ed.* **2002**, *41*, 1468.
[59] B. Hartke, In Computational Molecular Science; P. R. Schreiner, W. D. Allen, M. Orozco, W. Thiel, P. Willett, Eds.; Wiley Interdisciplinary Reviews, WIREs Comput. Mol. Sci. **2011**, *1*, 879.
[60] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, Numerical Recipes – The Art of Scientific Computing, 2nd ed.; Cambridge University Press, Cambridge, UK, **2003**.
[61] M. J. D. Powell, *IMA J. Numer. Anal.* **2008**, *28*, 649.
[62] M. J. D. Powell, Report No. DAMTP 2009/NA06, Centre for Mathematical Sciences, University of Cambridge, Cambridge, UK.
[63] A. C. T. van Duin, W. A. Goddard, III, A. L. Yakovlev, ReaxFF **2012**, SCM, Theoretical Chemistry, Vrije Universiteit, Amsterdam, The Netherlands, Available at: http://www.scm.com/, last accessed: 6 July 2013.
[64] W. Gropp, E. Lusk, A. Skjellum, Using MPI, MIT Press: Cambridge, MA, 1999.
[65] M. R. LaBrosse, J. K. Johnson, A. C. T. van Duin, *J. Phys. Chem. A* **2010**, *114*, 5855.

[66] A. C. T. van Duin, A. Strachan, S. Stewman, Q. Zhang, X. Xu, W. A. Goddard, III., *J. Phys. Chem. A* **2003**, *107*, 3803.

[67] J. C. Fogarty, H. M. Aktulga, A. Y. Grama, A. C. T. van Duin, S. A. Pandit, *J. Phys. Chem. A* **2010**, *132*, 174704.

[68] C. A. Brewer, Available at: http://www.ColorBrewer2.org, **2012**, last accessed: 6 July 2013.

[69] T. Weise, Global Optimization Algorithms – Theory and Application, Available at: http://www.it-weise.de/, **2011**, last accessed: 6 July 2013.

[70] L. Davis, Ed. Genetic Algorithms and Simulated Annealing, Pitman Publishing: London, **1987**.

[71] L. Davis, Ed. Handbook of Genetic Algorithms, Van Nostrand Reinhold: New York, **1991**.

[72] L. Spector, Ed. Gecco-2001, In Proceedings of the Genetic and Evolutionary Computation Conference; Morgan Kaufmann Publishers: San Francisco, **2001**.

[73] J. Lässig, K. H. Hoffmann, *Phys. Rev. E* **2009**, *79*, 046702.

[74] H. R. Larsson, BSc thesis, University of Kiel, May/June **2012**.

[75] O. Rahaman, A. C. T. van Duin, W. A. Goddard, III, D. J. Doren, *J. Phys. Chem. B* **2011**, *115*, 249.

[76] K. Michaelian, *Chem. Phys. Lett.* **1998**, *293*, 202.

Received: 16 April 2013
Revised: 17 June 2013
Accepted: 24 June 2013
Published online on 15 July 2013