ReaxFF Parameter Optimization with Monte Carlo and Evolutionary Algorithms: Guidelines and Insights

Ganna Shchygol,[†,‡] Alexei Yakovlev,[‡] Tomáš Trnka,[‡] Adri C.T. van Duin,[¶] and Toon Verstraelen[*,†]

†*Center for Molecular Modeling (CMM), Ghent University, Technologiepark-Zwijnaarde 46, B-9052, Ghent, Belgium*

‡*Software for Chemistry & Materials (SCM) B.V., De Boelelaan 1083, 1081 HV Amsterdam, The Netherlands*

¶*Department of Mechanical and Nuclear Engineering, The Pennsylvania State University, University Park, PA 16802, United States*

E-mail: Toon.Verstraelen@UGent.be

Abstract

ReaxFF is a computationally efficient force field to simulate complex reactive dynamics in extended molecular models with diverse chemistries, if reliable force-field parameters are available for the chemistry of interest. If not, they must be optimized by minimizing the error ReaxFF makes on a relevant training set. Because this optimization is far from trivial, many methods, in particular genetic algorithms (GAs), have been developed to search for the global optimum in parameter space. Recently, two alternative parameter calibration techniques were proposed, i.e. Monte-Carlo Force Field optimizer (MCFF) and Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES). In this work, CMA-ES, MCFF and a GA method (OGOLEM) are systematically

compared using three training sets from the literature. GA shows the smallest risk of getting trapped into a local minimum, whereas CMA-ES is capable of reaching the lowest errors for two third of the cases. For each method, we provide reasonable default settings and our analysis offers useful guidelines for their usage in future work. An important side effect impairing the parameter optimization is numerical noise. A detailed analysis reveals that it can be reduced, e.g. by using exclusively unambiguous geometry optimizations in the training set. Even without this noise, many distinct near-optimal parameter vectors can be found, which opens new avenues for improving the training set and detecting overfitting artifacts.

1 Introduction

Molecular dynamics (MD) is a powerful tool to study the temporal evolution of various atomistic models under realistic conditions. An essential ingredient in such simulations is a computationally efficient method to compute reliable atomic forces at every time step of an MD simulation. The quantum-mechanical (QM) treatment of the molecular electronic wavefunction allows one to compute these forces for any (reactive) chemical system, e.g. using density functional theory (DFT) methods. However, for long molecular dynamics simulations (nano- to microseconds) of extended atomistic models (up to millions of atoms), even DFT approximations become computationally prohibitive. Only linear-scaling and massively parallel (tight-binding) DFT implementations can handle such system sizes, by taking advantage of a large number of CPU or GPU cores, reducing the wall time of a single MD step to the order of minutes or hours.[1–6] Alternatively, one may use so-called force-field (FF) or molecular mechanics approximations, where the electronic structure calculation is replaced by a much cheaper and more approximate model to compute forces acting on atoms. Many force fields (FFs) model chemical bonds by simple springs with empirical parameters in Hooke's law, which immediately reveals their major drawback, i.e. most force fields cannot describe chemical reactions. Reactive FFs overcome this limitation with a more complex

mathematical expression that can describe reactive processes[7–10] and ReaxFF[11–14] is one of the most popular models in this category. ReaxFF owes its popularity to a combination of unique advantages. It can be reparameterized for different combinations of chemical elements, which makes ReaxFF broadly applicable. Furthermore, the superior computational efficiency of ReaxFF has been demonstrated in comparison to (tight-binding) DFT approximations[15] and neural-network potentials.[16] As opposed to hybrid QM/MM methods, ReaxFF can describe complex systems in which many reactive events occur simultaneously throughout the atomistic model, such as simulations of hydrocarbon oxidation[17,18] or mechanical wear resistance of graphene.[19]

Even though a detailed description of the complete mathematical form of ReaxFF goes beyond the scope of this paper, it is instructive to review a few of its essential aspects. The ReaxFF model is a potential energy expression for an atomistic model that ultimately takes the Cartesian coordinates of atomic nuclei and a set of empirical ReaxFF parameters as input. Analytic differentiation of this energy with respect to atomic positions yields the forces needed in a molecular dynamics simulation. The ReaxFF energy is a sum of many contributions,

$$E_{\mathrm{ReaxFF}} = E_{\mathrm{bond}} + E_{\mathrm{over}} + E_{\mathrm{under}} + E_{\mathrm{val}} + E_{\mathrm{tors}} + E_{\mathrm{vdW}} + E_{\mathrm{charge}} + E_{\mathrm{specific}} \qquad (1)$$

most of which are covalent terms responsible for describing local chemical phenomena: bond breaking & formation (bond), over-coordination (over), under-coordination (under), valence angle bending (val), bond torsion (tors). The next two terms describe non-covalent interactions between all pairs of atoms within a cutoff distance, even when they are not chemically bonded. The van der Waals (vdw) interaction is similar to the Morse potential and captures any effects due to steric repulsion and dispersion interactions. The energy of the fluctuating charge model (charge) includes the pairwise screened Coulomb interaction and the polarization cost of the fluctuating charges. The local energy terms depend on bond orders,

calculated for every pair of nearby atoms from their interatomic distances. Because these bond orders are recomputed for every atomic configuration, ReaxFF is capable of describing chemical phenomena. Finally, ReaxFF contains several optional terms, which are not employed in all of the parameterizations:

$$E_{\text{specific}} = E_{\text{lp}} + E_{\text{pen}} + E_{\text{coa}} + E_{\text{conj}} + E_{\text{trip}} + E_{\text{H-bond}} + E_{\text{lg}} + \dots \tag{2}$$

These terms can handle phenomena that are only of interest in specific cases: interaction with a lone pair (lp), penalty for valence angles between two double bonds (pen), conjugation correction for valence angles (coa), conjugation correction for bond torsions (conj), triple-bond stabilization (trip) and hydrogen bonding (H-bond). The low-gradient (lg) pairwise R^{-6} term was introduced to better describe long-range dispersion interactions[20] but is rarely used in recent parameterizations. More specific corrections exist, not listed explicitly in Eq. (2) because they are irrelevant for this work, such as the C2 (carbon dimer) energy term[21] or the iron dimer term.[22] Another recent extension developed by van Duin is eReaxFF, which introduces explicit electron or hole particles that can interact with the atoms.[23]

As a consequence of its wide adoption, several ReaxFF implementations were developed next to the original "Standalone ReaxFF" by van Duin.[11,14,17] The development of new energy terms in ReaxFF were done in the original code and were later adopted in other implementations. One of the earliest parallel versions was presented in LAMMPS.[24] Other notable implementations of ReaxFF can be found in GULP,[25] the code by Nomura[26] and in PuReMD.[27] The latter two strongly focus on efficient parallelization on high-performance clusters. One concern with various implementations is that they are not fully compatible with the code by van Duin, e.g. because some were re-implemented from scratch and introduced modifications to make the potential energy surface smoother.[25] This introduces the risk that ReaxFF parameterizations from the literature, calibrated with one implementation, may no longer yield sensible results in another one. All results in this work were obtained

with ReaxFF in ADF2018, unless noted otherwise. This implementation is directly based on the original code by van Duin to assure that the functional form of the potential energy closely follows the original, but has an improved efficiency and parallel scaling.[28] This implementation intends to maintain a compatible potential energy with the standalone ReaxFF, but we did fix several bugs in the force evaluation to make geometry optimization more robust, in the frame of this project. While such changes introduce a slight incompatibility, we give preference to accurate forces.

The ReaxFF energy expression contains many empirical parameters, which need to be optimized before ReaxFF can be used for production simulations. Even though a considerable number of tuned parameter sets are published in the literature,[11,17,21,29–35] one must extend this effort if no parameters are available yet for the chemistry of interest. To find the new parameters, a training set must be constructed, which consists of reference properties, $x_{i,\mathrm{ref}}$, of molecules or crystals relevant to the chemistry of interest. The ReaxFF predictions for these properties, $x_{i,\mathrm{calc}}(\{p_j\})$, are determined by a parameter vector, p_j, which can be adjusted to minimize the deviation from the training data. To maintain compatibility with previous works, many of these parameters are constrained to historical values and on the order of 50 free parameters must be estimated. The quality of a parameter vector is quantified by an objective function, hereafter referred to as *the Error*. We used the same least-squares Error as in the original work by van Duin:[36]

$$\mathrm{Error}(\{p_j\}) = \sum_{i=1}^{n} \left(\frac{x_{i,calc}(\{p_j\}) - x_{i,ref}}{\sigma_i} \right)^2 \tag{3}$$

where the sum runs over all training data points. In each term, σ_i is an estimate of the acceptable deviations between the ReaxFF calculation and the corresponding reference value. In ADF2018, Error contributions take into account the periodicity of dihedral angles. For example, a calculated dihedral angle of $-170\,\mathrm{deg}$ and a reference value of $+170\,\mathrm{deg}$ results in only a difference of $20\,\mathrm{deg}$.

Finding the optimal parameters for a given training set is far from trivial. The traditional approach consisted of successive one-parameter parabolic extrapolations (SOPPE).[36] Even though the Error is a simple sum of squares, it is in practice an ill-behaved function of the parameters $\{p_j\}$, with significant numerical noise and many local minima, such that the SOPPE optimization becomes very laborious.[32,37] It seems more appropriate to employ global optimization algorithms. Brute-force global optimizers, which perform a grid search in the parameter space, are computationally not feasible because their cost scales exponentially with the number of parameters. Instead, several groups have designed genetic algorithms (GAs) specifically for ReaxFF parameterization, aiming at a good global-optimization efficiency.[34,37–39] It was shown that GAs can minimize the objective function equally well or even further than the SOPPE method originally introduced by van Duin.[37,40] Furthermore, these GAs no longer require manual intervention and human judgment while minimizing the Error. Next to GAs, many other techniques were proposed, such as a Multi-Objective Evolutionary Strategy (MOES),[41] a parallel local search algorithm,[42] Taguchi method based optimization,[43] Monte-Carlo force-field optimizer[32,44] (MCFF) and Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES).[45,46] The goal of this work is to assess and comprehend the efficiency of the MCFF and CMA-ES optimizers in comparison to a state-of-the-art GA. The tendency of optimizers to get trapped in a local minimum will be tested by restarting the fitting procedure from the same initial guess (with different random seeds) or from different random initial parameter vectors. Since GAs are also popular for the parametrization of other force fields,[47–53] we expect that the comparison in this work is also useful beyond the scope of ReaxFF.

Our comparison follows a relatively conservative approach to obtain representative test results for the ReaxFF community. The training sets from previous GA studies of Hartke were used without modifications,[37] except for minimal corrections to one set, as will be explained below. Furthermore, reasonable default settings were used for all optimization algorithms, without tuning them for each specific case. Further refinements for each com-

bination of training set and initial guess can improve the performance, for which ample suggestions are provided through the analysis of our results.

When simply minimizing the Error in Eq. (3), there is a significant risk to overfit the parameters. Overfitted parameters have a low Error for the training set but still produce unphysical results in realistic ReaxFF molecular dynamics simulations.[54,55] While we recognize the importance of this problem, this work mainly focuses on testing and understanding the strengths and weaknesses of the parameter optimization algorithms. Nevertheless, to avoid overfitting, one may introduce a so-called test set, in addition to the training set. An optimization can be interrupted when the Error on the test data increases, even though the Error on the training data is still decreasing, a technique commonly referred to as "early stopping".[34] Overfitting can also be observed by unrealistic values of optimized parameters for which prior knowledge is available, e.g. an atomic radius. In such cases, it could be beneficial to fix the parameter or to limit its interval of allowed values.

The remainder of this paper is structured as follows. In section 2, we describe training sets, ReaxFF geometry optimization details, parameter optimization algorithms with their settings and the evaluation criteria used to compare the performance of these algorithms. The results of this comparison are described in section 3, where we also discuss guidelines for an effective usage of the optimization algorithms and good practices for the design of a new training set. Finally, section 4 summarizes the main conclusions and gives an outlook on future work.

2 Methods

2.1 Training sets

Three training sets from the literature are used in this work: one for solid and liquid cobalt and defects in cobalt crystals,[31] one for silica clusters and (porous) crystals[30] and one focusing on disulfide mechanochemistry.[34] Key properties of the three training sets are compared in

Table 1, highlighting fundamental differences in the types of information they contain. For example, the Cobalt set only contains energy data, while the Disulfide set is the only one containing atomic forces. Moreover, these training sets also differ in the number of free parameters.

Table 1: Overview of ReaxFF training sets used in this work. The number of data points in each training set is broken down into five categories: C (atomic charges), G (geometry, internal coordinates), F (Cartesian atomic forces), P (cell parameters) and E (reaction energies). Note that unused geometries in the `geo` files were not counted and similarly non-existing geometries in the `trainset.in` file were ignored. The Error of the optimal parameters reported in the literature is included, and Errors for the same parameters were recomputed with ADF.

Label	N_{par}	N_{geo}	Number of data points						Literature	ADF Error	
			C	G	F	P	E	Total	Error	Default	Torsion2013
Cobalt	12	146					144	144	1444	1459	1459
Silica	67	302	5	26		13	265	309	3196	4607 *	6438 *
Disulfide	87	231		255	4401		219	4875	12393 7574 †	15577	16271

* Computed with a development version of ADF2019.3 instead of ADF2018.
† The lower Error value could be reached by overfitting the paremeters.[34]

In the first place, a training set provides *primary* information to evaluate the Error, see Eq. (3). This primary information consists of molecule or crystal geometries for which reference data is provided, including atomic charges, equilibrium geometries (internal coordinates), Cartesian atomic forces, cell parameters and reaction energies. The original papers from which the training sets were taken, also contain *secondary* information: the final optimal parameters and constraints that were imposed during the parameter optimization. The first kind of constraint is the reuse of existing ReaxFF parameters without modification to maintain backward compatibility. This also facilitates the generation of a new training set, because it only needs to contain data to determine the new parameters. The second type of constraint is the allowed interval, further denoted as $[p_i^{\min}, p_i^{\max}]$, for each parameter p_i. We followed these constraints as closely as possible. However, for the Silica training set, some of these intervals had to be modified, as explained in section S1 of the supporting information.

Despite the fact that we used training sets from the literature without modification to their primary information, except for one typographical error described in section S1 of the supporting information, we encountered difficulties reproducing literature values of the Error for optimal parameters published with these training sets. Table 1 compares Errors reported in the literature with the ones we computed with ADF (Default settings). While the correspondence is acceptable for the Cobalt training set, the Error computed with ADF for the Silica and Disulfide training sets is significantly larger. As will be explained in section 3, the Error function for these two training sets is inherently non-robust because of ill-behaved geometry optimizations. For a small number of molecules, the optimal geometry is extremely sensitive to irrelevant details. For example, a change in parameters by less than 0.5% can result in a different conformation, causing changes of the Error by 1000 units or more. This high sensitivity makes it practically impossible to reproduce an Error value from the literature. Note that we have computed the Error for the Silica training set with a development version ADF2019.3, because older versions incorrectly parsed a negative van der Waals well depth parameter. The newer version of ADF was only needed for these two Errors because all other calculations employed positive parameters for the van der Waals well depth.

2.2 Geometry optimization

Properties of the optimized geometries are used to evaluate the Error [see Eq. (3)], except when energies or forces of non-equilibrium structures are specified in the training set. Hence, many geometries need to be re-optimized for any new trial parameter vector during the parameter optimization.

We have taken several measures to improve the geometry convergence because it results in a smoother Error function, which facilitates the parameter fitting. First of all, the line-search algorithm in the L-BFGS optimizer was modified to handle cases with negative curvature more gracefully. Secondly, we used relatively stringent convergence settings: the

geometry optimization continues until the maximum force on any of the atoms drops below 0.1 kcal mol^{-1} Å^{-1}. In some problematic cases, 3000 optimization steps were not sufficient, after which the algorithm stopped, and the last geometry was used instead of the optimal one. Such convergence issues may appear when the optimal geometry is near a point with discontinuous ReaxFF forces or an ill-conditioned Hessian matrix, representing the curvature of the potential energy surface. Finally, the Torsion2013 option was implemented to reduce discontinuities in the interatomic forces, as explained in section S2 of the Supporting Information. This option has only a small effect on the potential energy landscape and ReaxFF results in general. For example, the last two columns of Table 1 show that the Torsion2013 correction has only a modest impact on the Error value. (The error for Cobalt is not affected because this ReaxFF parameterization does not use torsional terms.) Because this option improves geometry convergence with minimal side effects, it was enabled in all parameter optimizations.

2.3 Initial guess of the parameter vector

Initial guesses of different quality were generated to test the influence on the outcome of the parameter optimization. The 'best' type of initial guess is the optimal parameter vector previously reported for these training sets. However, the literature parameters for the Silica training set were unsuitable as an initial guess because nine parameters exceeded their allowed intervals. In this case, we used minimally corrected parameters as best guess instead. (See section S1 of the supporting information for details.) In general, the best guess may not be optimal in this work for two reasons: (i) the Error function has reproducibility issues as explained above and (ii) the literature parameters for the Disulfide training set were obtained with early stopping.

An 'educated' initial guess was constructed from a database of ReaxFF parameters maintained by SCM,[56] using only parameters published prior to the corresponding training set. For every parameter, the following steps were taken:

1. All unique historical parameter values were looked up for the same type of parameter, associated with the same combination of chemical elements. Only those lying in the allowed interval for the training set, $[p_i^{\min}, p_i^{\max}]$, were retained and their median was used as a guess.

2. If in the previous step, no historical parameters were found in the allowed interval, the search for unique historical parameters was extended and they only had to be of the same type but they were allowed to be associated with other chemical elements. Of all these values, we took the median.

3. If again no historical values could be found in the previous step, we just took the center of the allowed interval as initial guess.

In case of the Cobalt training set, there is only one parameterization predating the training set[21] and the result of the above procedure is that the educated guess for Cobalt coincides with the parameters from Ref. 21.

Finally, 10 'random' initial parameter vectors were constructed for each training set. For every parameter, we sampled random values from a uniform distribution over the interval $[p_i^{\min} + (p_i^{\max} - p_i^{\min})/4, p_i^{\max} - (p_i^{\max} - p_i^{\min})/4]$, i.e. the central segment of the allowed interval spanning half the width.

Other types of prior knowledge could be used to construct an initial guess, e.g. certain ReaxFF parameters correspond to properties of chemical elements or bonds, for which experimental or ab initio data is available. However, this approach is only applicable to parameters with a physical interpretation and we have not attempted to construct such guesses in this work.

2.4 Parameter optimization algorithms

The settings of the parameter optimization algorithms described in this section are not excessively tuned for each combination of training set and initial guess. Instead, our settings

are reasonable defaults with which we aimed at an acceptable trade-off between global and local optimization performance for one run. With this choice, we obtain results that a non-expert can expect. While case-by-case tuning of optimization algorithm settings can be beneficial, subsequent tweaking of settings becomes computationally infeasible when the cost of a single run is already high.

The first algorithm in our comparison, the Monte-Carlo Force Field (MCFF) optimizer,[32] is primarily inspired by a physical model. By sufficiently slowly cooling down a many-particle system, it will eventually reach its ground state. By applying the same process to a simulated system, one may obtain a good approximation of the global minimum of any high-dimensional function, a technique commonly referred to as *simulated annealing*.[44] The two remaining optimization algorithms are Evolutionary Algorithms and are inspired by the biological model of evolution: a population, where individuals are defined by their genes (parameter vectors), is evaluated and only the fittest individuals (lowest Error) are retained (in a modified form) in the next generation. Over many generations, the genes evolve towards optimal fitness. Evolutionary Algorithms form a class of parameter optimization algorithms that simulate this mechanism, with the same purpose of finding the minimum of a high-dimensional function. Two algorithms from this class are considered in this work: the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES)[45,57] and a Genetic Algorithm (GA) implemented in OGOLEM.[37]

2.4.1 Monte-Carlo Force Field (MCFF) optimizer

MCFF uses simulated annealing to find optimal parameters.[32,44] While this is in principle a global optimization method, a true global optimization with simulated annealing requires very slow cooling rates, which can be too costly in practice. This difficulty is comparable to rapid cooling of a liquid, which can result in a glass instead of a crystal with a lower energy.

At every iteration, MCFF makes a small change to the parameter vector and computes the corresponding change in the Error function. In this work, the small step consisted of

a random change, of 10% of the parameters, sampled from a uniform distribution over the interval $[-(p_i^{\max}-p_i^{\min})s/100, (p_i^{\max}-p_i^{\min})s/100]$, where s controls the magnitude of step size. When the error decreases, the step is always accepted. In case of an increase, it is accepted with a probability $\exp(-\beta_n \Delta\text{Error})$, where β_n is the inverse dimensionless temperature at iteration n. The initial value of s was set to 1.0 and MCFF dynamically updates s to keep the acceptance ratio within user-specified bounds, in our case $[50\%, 70\%]$.

For every combination of training set and initial guess of the parameters, we performed three MCFF runs with 9k, 3x3k and 45k iterations. In case of 3x3k, 3 MCFF runs of 3000 steps were done in series, where the second and the third run are restarts using the optimal parameters from the previous run as initial guess.[16] The initial inverse temperature was always determined by

$$\beta_0 = \sqrt{\frac{N_{\text{par}}}{2}} \frac{1}{C_1 \text{Error}_0}, \tag{4}$$

where N_{par} is the number of optimized parameters, Error_0 is the Error of the initial parameters. In case of the second and third segment of a 3x3k run, β_0 is derived from the initial Error of the current restart instead of the Error of the initial guess. C_1 is approximately the initial magnitude of relative thermal fluctuations of the Error and was set to 1, allowing MCFF can escape local minima easily. If one prefers MCFF to perform a more local search, C_1 should be reduced by one or two orders of magnitude. The final inverse temperature is set to

$$\beta_N = \sqrt{\frac{N_{\text{par}}}{2}} \frac{1}{C_2}, \tag{5}$$

where C_2 is approximately the absolute fluctuation of the error at the final iterations and was set to 5. Such a small value for C_2 will let MCFF converge to the bottom of a (local) minimum. The above relation between Error fluctuations and inverse temperature would be exact if the Error were a quadratic function of the ReaxFF parameters and the sampling were complete, as shown in section S3 of the supporting information. In practice, these relations are approximate because the Error is a more intricate function and the number of

steps is too small for a complete sampling, especially when the Error is nearly flat in some directions.[38] To obtain an annealing simulation, β_n is divided by a constant factor at every iteration of the MCFF algorithm:

$$\frac{\beta_n}{\beta_{n+1}} = \sqrt[N]{\frac{C_2}{C_1 \mathrm{Error}_0}} \tag{6}$$

where N is the total number of iterations. The above configuration of the MCFF algorithm can be implemented with the control parameters in Table 2.

Table 2: MCFF and CMA-ES settings used in the ReaxFF control file. Control parameters not included in the table are left to their default value.

MCFF		
`mcffit`	$N = 9000$, 3000 or 45000	Number of MCFF iterations.
`mcbeta`	$\beta_0 = \sqrt{\frac{N_{\mathrm{par}}}{2}} \frac{1}{C_1 \mathrm{Error}_0}$	Initial inverse temperature.
`mcbsca`	$\frac{\beta_n}{\beta_{n+1}} = \sqrt[N]{\frac{C_2}{C_1 \mathrm{Error}_0}}$	Value by which the inverse temperature is divided at every step.
`mcstep`	1.0	The initial value for the step size s. The maximum change of a parameter in one MC step is `mcstep`/`mcrxdd`.
`mcrxdd`	100	Constant denominator in the expression for generating random steps. (In principle redundant as it can be absorbed into s.)
`mcscps`	1.05	The step size s is multiplied (divided) by this factor when the acceptance ratio is too high (low).
`mctart`	50.0	The target (minimum) acceptance ratio
`mcmart`	70.0	The maximum acceptance ratio
`mcacpf`	0.1	Fraction of the parameters that is changed in one MCFF step.
CMA-ES		
`mcffit`	$N = 20000$	Maximum number of CMA-ES iterations.
`ffotol`	`TolX` $= 10^{-6}$ or 10^{-5}	CMA-ES convergence criterion. (See text for details.)
`mcrxdd`	$N_\sigma = 4$	Controls the width of the initial normal distribution in parameter space.

2.4.2 Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES)

CMA-ES is a stochastic gradient-free optimization algorithm proposed by Hansen *et al*[45,57] and is occasionally used for force-field parametrizations.[46,53,58,59] Starting from a user-provided initial guess, CMA-ES iteratively improves a multivariate normal distribution in the parameter space to find a distribution whose random samples minimize the objective function. In essence, one iteration consists of the following steps and we refer to Ref. 45 for a detailed description:

1. A population of $\lambda = 4 + \lfloor 3 \ln N_{\mathrm{par}} \rfloor$ random points (trial parameter vectors) is drawn from the normal distribution, where N_{par} is the number of parameters being optimized. The non-elitist version of the algorithm was used, denoted as $(\mu/\mu_W, \lambda)$-CMA-ES, implying that no parameter vectors from previous iterations were added to the population.

2. The Error is computed for all of the trial parameter vectors.

3. The population is sorted by increasing Error and only the first $\lambda/2$ points are retained and assigned a weight, according to the default logarithmic weighting scheme from Ref. 45.

4. The mean (center) and covariance of the normal distribution are updated using the weighted points, using heuristic rules explained in Ref. 45.

5. When $\sigma \|\mathbf{p}_g\|$ and $\sigma \max_i \sqrt{C_{ii}}$ drop below a threshold, `TolX`, convergence is reached and the algorithm stops. In these criteria, σ is a variable step size, C is the current estimate of the covariance matrix and $\mathbf{p}_g$ is an average over previous steps with an exponential window. More details on these quantities can be found in Ref. 45. Alternatively, one may also stop after a maximum number of iterations.

The value of the objective function at each trial point is only used to rank the points, which makes CMA-ES invariant to any rank-preserving (strictly increasing) transformation of the

objective function. As the evaluation of each trial point is completely independent from the rest of the population, this step of CMA-ES is trivially parallel.

Starting from the initial guess, the covariance matrix C is incrementally improved by the feedback from sampled points and tends to approximate the inverse Hessian matrix, thus capturing the relative sensitivities of the parameters and also the correlations between them. The step length σ (overall width of the sampled distribution) is automatically controlled by the algorithm in every iteration, depending on the directions of previous steps. If subsequent steps move in a similar direction, the step length is increased accordingly. If, instead, subsequent steps tend to be in opposite directions, the algorithm is overshooting an optimum and thus responds by scaling down the step length.

The initial mean of the normal distribution is set to the initial guess (see section 2.3) and the initial covariance matrix is diagonal with each diagonal element set to $((p_i^{\max} - p_i^{\min})/N_\sigma)^2$, where $N_\sigma = 4$. With this value of N_σ, CMA-ES starts with a relatively broad initial distribution, such that the algorithm explores a large portion of the parameter space before converging. One may turn CMA-ES into a more local optimizer with higher values of N_σ. We used `TolX` $= 10^{-6}$ for the Cobalt training set and `TolX` $= 10^{-5}$ for the two other training sets. These convergence criteria are very tight, which allows us to test if the algorithm can find better solutions even after it has practically converged. Finally, we also terminate CMA-ES when it reaches 20k steps. Our configuration of the CMA-ES algorithm can be reproduced with the control parameters in Table 2.

The CMA-ES algorithm is implemented using the `c-cma-es` library.[60] The upper and lower bounds on the parameters ($p_i^{\min}$ and $p_i^{\max}$) are imposed by setting the Error to the maximum floating point value whenever parameters fall outside their allowed interval.

Because multiple Error evaluations are used in one CMA-ES iteration (λ defined above), one must be careful when comparing the efficiency with MCFF, in terms of number of iterations needed to achieve a low Error. For the Cobalt, Silica and Disulfide training sets, λ is 11, 16 and 17, respectively. In a serial calculation, this would be the relative cost of one

CMA-ES iteration compared to MCFF. However, in ADF, multiple Error evaluations can be carried out in parallel within one CMA-ES iteration. To avoid ambiguities with differences in computational cost of one iteration in different algorithms, the number of Error evaluations (#Error calls) will be used to quantify the computational cost.

2.4.3 Genetic Algorithm (GA)

Compared to CMA-ES, a genetic algorithm follows Darwin's theory of evolution more closely. GAs mathematically model the crossover and mutation of genes upon reproduction and the survival of the fittest species in each generation. In the context of ReaxFF, GAs have been proposed as effective global parameter optimization strategies.[34,37–39] To remain consistent with the previous GA studies for optimizing ReaxFF parameters, the OGOLEM implementation was selected for the current work.[34,37]

In a genetic algorithm, a parameter vector is denoted as a set of *genes* of an *individual* and a *population* of such individuals is optimized by constructing child individuals from which only the fittest are retained. In this work, a starting population of 500 individuals comprises an initial guess (see section 2.3) and a set of uniformly distributed vectors within the allowed parameter intervals. In OGOLEM's pool-based algorithm, two new children are generated and evaluated in one iteration and the best one replaces a lesser-fit individual from the pool as soon as the calculation of the fitness has completed. Any new child is derived from two parents, selected from the population, with a higher probability of selecting fitter ones. The parents' genes are exposed to either binary or unary genetic operations to produce genes for the new individual. Because the genes of fitter parents contribute more to the following generations, a genetic algorithm produces a fit population after many iterations. In our work, fitter means a lower value of the Error in Eq. (3).

The complete input files used for OGOLEM are provided in section S4 of the supporting information and only the most important settings are summarized below. The number of iterations is set to 110k, 160k and 170k for the Cobalt, Silica and Disulfide training

sets, respectively. This way, the maximum number of Error evaluations is the same as for CMA-ES. A child's genes are derived using either a binary (recombination or crossover) or a unary (mutation) operator, with probabilities of 80% and 20%, respectively. Binary recombination implies an exchange of genetic information between parents' genes and can be realized with different crossover flavors. In this work, we used (i) a mixing recombination operator (20% probability), which determines child genes as a weighted average of its parents' values and (ii) a multipoint exchange (80%), which defines a random number of cutting points in the parameter vector and switches between the two parents genes after each point. The unary mutation generates new child's genes by making random modifications to the genes of one parent. Either a subset of the parameters are taken from a uniform distribution over the entire interval of allowed values (20%) or small perturbation to some parameters are sampled from a Gaussian distribution (80%). OGOLEM also supports niching, which defines equidistant bins for every parameter and imposes conditions on the number of individuals that may occupy each bin. We used the niching setting from Ref. 37, but these had a relatively low impact on our calculations. The number of individuals rejected by the niching conditions was marginal in all our calculations.

In addition to the basic settings described above, more details can be controlled through the OGOLEM input file (see section S4 of the supporting information), which is both a weakness and a strength of genetic algorithms. Ample settings allow one to tune the algorithm to specific use cases, but they also make it far from trivial for casual users to control these settings effectively.

A particular advantage of the pool-based GA implementation in OGOLEM is its parallel efficiency. CMA-ES supports synchronous parallelization, which means that all Error calculations within one iteration have to be completed before a new iteration can be started. When one Error calculation takes much longer than all others, some cores become idle until the last calculation of an Error within one iteration has completed. The pool-based GA allows for asynchronous parallelization, meaning that whenever an Error calculation has completed,

the algorithm can decide which parameter vector to process next, without having to wait for the completion of other Error calculations, effectively avoiding idle CPU cores.

2.5 Evaluation criteria

While the main performance criterion in this work is the lowest value of the Error reached during an optimization, we also check other aspects of the trajectories through parameter space followed by the optimization algorithms. For each combination of training set, initial guess and optimizer settings, we performed 10 optimizations. All optimization runs are ranked by the lowest Error reached at any point during the optimization. The lowest Error for the best, second best and worst run are compared to test (i) the sensitivity to the initial guess and (ii) the tendency of the algorithm to get locked into local minima. We also report the number of Error evaluations needed to reach this lowest Error for the best run, which is a good indication for efficiency of the optimization. To asses how much the parameters have changed, we also compute the distance (Euclidean norm) d_1 between the initial guess and the parameter vector of the lowest Error (for the best run) in reduced parameter units:

$$\tilde{p_i} = \frac{p_i}{p_i^{\max} - p_i^{\min}} \tag{7}$$

Finally, we also compute the distance d_2 between the optimal parameters of the best and second best run, to investigate parameter degeneracies. It was previously observed that the Error is not sensitive to certain linear combinations of parameters, making the optimal parameters degenerate.[38]

3 Results and discussion

In total 45 sets, with 10 parameter optimizations each, were carried out in this study. There is one set for every combination of (i) three different training sets (Cobalt, Silica and Disulfide), (ii) three optimizers (MCFF, CMA-ES and GA) with three different settings for MCFF and

(iii) three different qualities of the initial guess (best, educated and random). A summary of the numerical results for each set, in line with the evaluation criteria discussed above, is given in Tables 3, 4 and 5. In addition, Figure 1 shows the decrease of the error as function of the number of Error evaluations for four out of 45 representative sets. Similar plots for all 45 sets are included in section S5 of the supporting information. In case of MCFF, all Error values are plotted, for CMA-ES, only the lowest Error at each iteration is shown and for GA, the lowest Error up to a given point is depicted.

The most important result in Tables 3, 4 and 5 is the significant spread on the lowest Error between the best and the worst out of 10 runs, for all 45 combinations of algorithm, initial guess and training set. The variation is explained by the random seed affecting stochastic optimization algorithms. When random initial guesses are used within one set, these differences also affect the outcome. For the random and educated guesses, the lowest Error in the worst run is always significantly higher compared to the Error values from the literature in Table 1. Hence, performing just a single run with any of the algorithms from this work, starting from a realistic guess, holds a significant risk for obtaining relatively poor parameters.

The spread between best and worst run is a good inverse measure for the robustness of an optimization algorithm, i.e. it shows how reproducible the entire procedure is. From this point of view the GA results are the most appealing, with the lowest spread. The least robust optimizations are the MCFF-9k runs, which have a significant risk of getting trapped in high local minima of the Error function. This is partially addressed by restarting the MCFF a few times (3x3k) or by annealing slowly (45k), but this is still, especially for random initial guesses, not as robust as GA. CMA-ES is generally more robust than MCFF, but it is still more sensitive to initial guess and random seed than the GA, in particular for the silica training set.

A second important observation is that for the Cobalt and Disulfide training sets, the lowest Error from the best (out of 10) CMA-ES runs is significantly lower than that of

Table 3: Overview of Cobalt parameter optimization results. Each row summarizes 10 optimization runs carried out with identical settings. (See text for details.)

Algorithm (#Error calls)	Guess	Best run		2^{nd}-best run	Worst run	d_1 [2*]	d_2 [3*]
		#Error to lowest [k] [1]	Lowest Error	Lowest Error	Lowest Error		
MCFF-9k (9k)	best	9	1386	1392	1477	0.15	0.18
	edu	8	1737	2066	3925	0.81	0.84
	rand	9	1623	2430	4492	1.05	1.25
MCFF-3x3k (9k)	best	6	1362	1374	1450	0.15	0.17
	edu	6	1711	1860	3439	0.63	0.37
	rand	7	1708	1842	6306	0.34	0.80
MCFF-45k (45k)	best	39	1360	1363	1449	0.22	0.25
	edu	41	1532	1806	3265	1.04	1.32
	rand	45	1422	1458	3064	1.10	1.12
CMA-ES-20k ($\leq$220k)	best	50	1180	1199	2437	1.45	0.43
	edu	89	1157	1172	2467	1.47	0.99
	rand	23	1150	1168	3013	1.01	0.70
GA-110k (220k)	best	217	1344	1357	2468	0.68	0.78
	edu	220	1347	1375	2845	1.10	0.72
	rand	220	1346	1468	2765	1.19	0.58

[1] The number of Error evaluations before the lowest Error was reached, divided by 1000.

[2] Distance from the initial guess to the optimal parameter vector for the best run.

[3] Distance between the optimal parameter vectors in the best and second best run.

[*] Distances are in dimensionless parameter units, see Eq. (7).

Table 4: Overview of Silica parameter optimization results. Each row summarizes 10 optimization runs carried out with identical settings. (See text for details.)

Algorithm (#Error calls)	Guess	Best run		2^{nd}-best run	Worst run	d_1 [1*]	d_2 [2*]
		#Error to lowest [k] [1]	Lowest Error	Lowest Error	Lowest Error		
MCFF-9k (9k)	best	4	5709	6096	8622	2.38	2.89
	edu	5	5995	6755	8125	2.48	2.82
	rand	7	5882	6334	11795	2.46	3.23
MCFF-3x3k (9k)	best	9	4639	4724	5825	2.34	2.58
	edu	6	4598	4678	6425	1.29	2.11
	rand	9	4574	6913	57189	2.63	2.76
MCFF-45k (45k)	best	17	4885	5034	7298	2.66	2.83
	edu	12	5632	5695	11412	2.56	2.93
	rand	36	5059	5536	9768	2.76	2.34
CMA-ES-20k ($\leq$320k)	best	95	3791	3890	10217	2.14	2.21
	edu	116	3742	3870	10539	2.08	2.40
	rand	209	3727	4097	7888	2.68	2.75
GA-160k (320k)	best	319	3587	3618	3986	2.15	2.37
	edu	215	3705	3712	4356	1.86	2.06
	rand	167	3577	3642	4062	2.55	2.57

[1] The number of Error evaluations before the lowest Error was reached, divided by 1000.

[2] Distance from the initial guess to the optimal parameter vector for the best run.

[3] Distance between the optimal parameter vectors in the best and second best run.

[*] Distances are in dimensionless parameter units, see Eq. (7).

Table 5: Overview of Disulfide parameter optimization results. Each row summarizes 10 optimization runs carried out with identical settings. (See text for details.)

Algorithm (#Error calls)	Guess	Best run		2^{nd}-best run	Worst run	d_1 [2*]	d_2 [3*]
		#Error to lowest [k] [1]	Lowest Error	Lowest Error	Lowest Error		
MCFF-9k (9k)	best	8	11899	11981	14305	1.80	2.49
	edu	8	14852	15721	19655	2.93	3.06
	rand	9	11960	14332	25634	2.75	2.77
MCFF-3x3k (9k)	best	9	10914	11248	13389	1.44	1.79
	edu	6	13754	14595	20816	2.65	2.79
	rand	6	13886	14311	39381	2.71	3.10
MCFF-45k (45k)	best	34	10605	11719	15341	2.83	3.18
	edu	44	9608	11828	19898	3.27	3.33
	rand	44	8507	9684	15274	2.90	2.69
CMA-ES-20k ($\leq$340k)	best	309	8994	10337	13257	2.34	2.89
	edu	248	8693	9128	15386	3.12	2.79
	rand	250	6716	9388	14665	3.10	2.57
GA-170k (340k)	best	328	18524	19213	22170	1.97	2.19
	edu	339	18054	18478	21270	2.44	2.46
	rand	340	19285	19489	21637	2.33	2.32

[1] The number of Error evaluations before the lowest Error was reached, divided by 1000.

[2] Distance from the initial guess to the optimal parameter vector for the best run.

[3] Distance between the optimal parameter vectors in the best and second best run.

[*] Distances are in dimensionless parameter units, see Eq. (7).

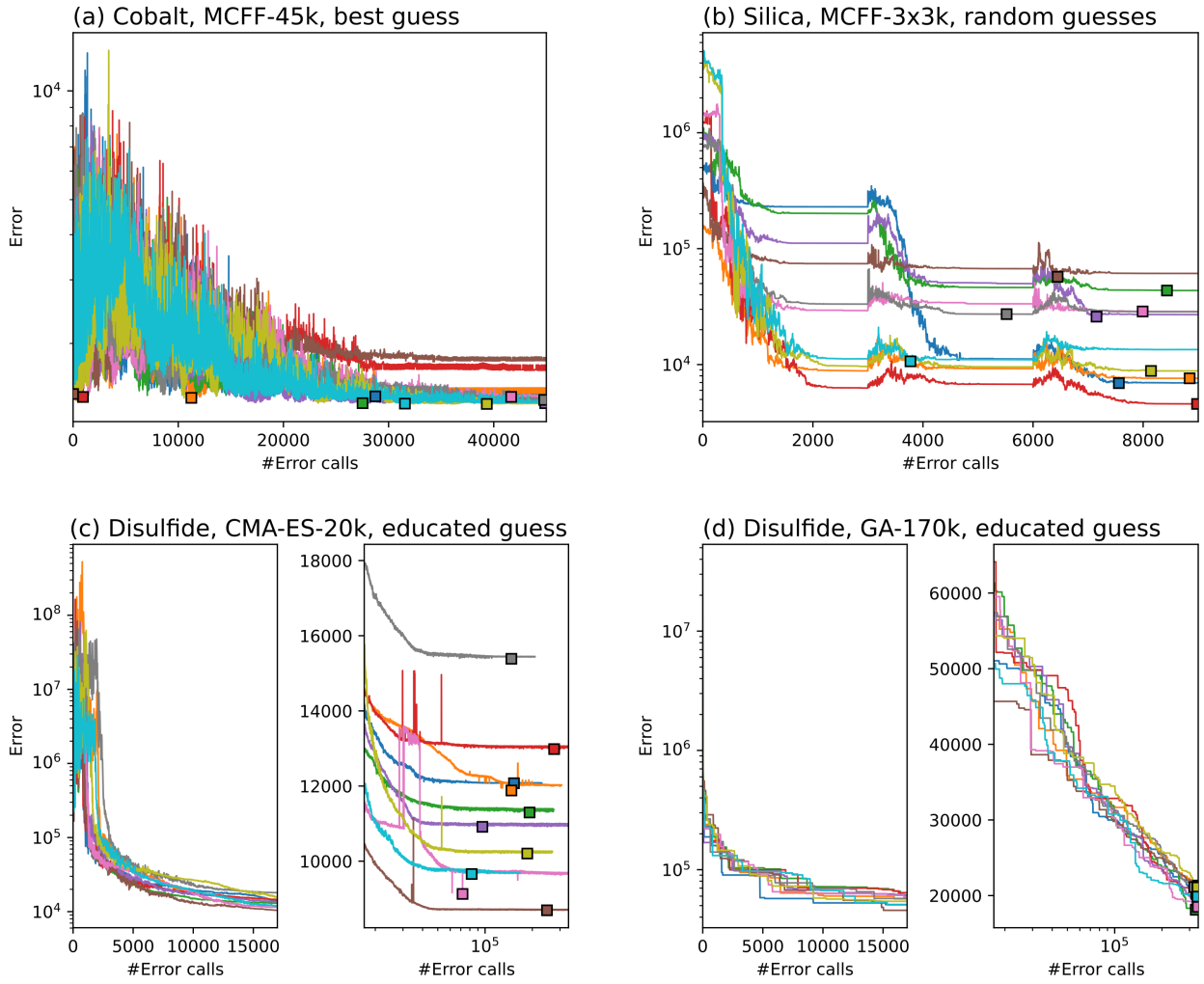

Figure 1: Representative plots of the Error as function of the number of Error calls, for MCFF (a, b), CMA-ES (c) and GA (d). Each plot displays the progress of the Error for 10 optimizations, with square boxes showing the lowest error achieved in each run. (Training set and algorithm settings are indicated in each panel.) Differences within one plot are caused by the stochastic behavior of the optimization algorithms and in panel (b) also by the differences between the random initial guesses. In panels (c) and (d), the plot on the right is a continuation of the plot on the left, with different axes to depict the Error near convergence.

the GA. With the current settings, the GA has not converged yet in all the cases. This is evident for the disulfide training set, i.e. no saturation has been reached yet in Fig. 1d and related figures on pages S19 and S21 of the Supporting Information. For the disulfide training set with best initial guess, the GA was continued up to 4 million iterations, which yielded a lowest Error in the best run of 9502, still without reaching a visible saturation of the Error. We expect even more GA iterations would further decrease the Error, which we have not tested due to limitations of our computational resources. CMA-ES is certainly not an exhaustive global optimizer, e.g. it may converge to a relatively high local minimum. However, it demonstrates remarkable local optimization efficiency because it exploits the parameter covariance,[45,57] unlike MCFF or GA.

The rate of convergence of the tested algorithms is also markedly different, as shown in Figure 1. MCFF slowly cools down the parameters and the lowest Error is usually encountered close to the end. More MCFF iterations (45k steps) led to lower Errors, as one would expect from any simulated annealing method. The restarts (3x3k steps) are in some cases advantageous over one slower annealing of 9k steps, see e.g. Table 4 for random initial guesses. Because the restarts do not dramatically deteriorate the lowest Errors for other cases, they can be used by default to reduce the risk of getting trapped into a local minimum with MCFF.[16] However, significant improvements are often made in the last restart, see e.g. Figure 1b, such that additional restarts could lead to even lower Error values. CMA-ES first explores a wide region of the parameter space, resulting in very high initial Errors. Afterwards, it exhibits a rapid decrease of the Error over the first few thousand Error calculations, after which the Error levels off to a plateau. In some rare cases, such a plateau is followed by another decrease in the Error after several 10k Error evaluations. Therefore, when using CMA-ES, it could be promising to implement a pruning scheme,[46] where one first performs a series of short CMA-ES optimizations with different initial guesses or random seeds and then continues only the most promising ones with additional CMA-ES iterations. The number of steps needed in the short runs depends on the complexity of the training set and the number

of free ReaxFF parameters. With hindsight, for the training sets considered in this work, a few thousand Error evaluations would have been sufficient. The progress of the Error of the GA follows a similar pattern as CMA-ES, starting with a very high values, followed by a rapid decrease. In comparison, CMA-ES reaches lower values earlier in most runs, with a few exceptions, confirming the above observations, i.e. that CMA-ES can be more efficient but is not as robust. Note that Fig. 1d and corresponding GA Error plots in section S5 of the supporting information show the lowest Error up to a given iteration. The GA continuously attempts to escape its current local minimum and many of these attempts result in a high Error, not shown in these plots.

The quality of the initial guess may have some positive influence on the lowest Error, but the effect is marginal. A (rare) logical example can be found when optimizing parameters with MCFF over 3x3k steps for the Disulfde set: the lowest Error with best, educated and random guesses are 10914, 13754 and 13886, respectively. However, the educated guess may also lead to inferior results in comparison to the random guesses. An interesting case is the Disulfide set with the CMA-ES optimizer, where the lowest Error with best, educated and random guess are 8994, 8693 and 6716, respectively. In addition, in all sets of 10 runs with random initial guesses, there is no significant rank correlation between Error for the initial guess and the lowest Error. The limited impact of the initial guess is consistent with our settings of MCFF, CMA-ES and GA. The algorithms will explore a significant part of the parameter space in the first iterations in an attempt to avoid convergence to a high local minimum. This also implies that the parameter trajectories quickly depart from the initial guess. One may lower the initial fluctuations in MCFF (lower C_1 in Eq. 4) or the initial width in CMA-ES (higher N_σ) to let these algorithms stay closer to the initial guess. Obviously, this also increases the risk of getting trapped into a local minimum with a high Error.

The performance of any of the selected parameter optimization algorithms, e.g. to find a lower Error, can be improved by carefully tuning the algorithm settings or by performing

subsequent runs, as shown in this work for MCFF. It would go beyond the scope of this work to perform such a tuning on each combination of training set, initial guess and optimization algorithm. Instead, our comparison shows the typical differences in behavior of the selected optimization algorithms, which a non-expert may expect using reasonable default settings.

Finally, we observed that two runs with the same initial guess and the same algorithm may converge to significantly different parameters, yet having nearly equal Error values. An illustrative example is the optimization of parameters for the Silica training set with CMA-ES using the educated initial guess. In this case, the best and second-best runs have similar Errors: 3742 and 3870, respectively. Yet, the distance between these two solutions ($d_2 = 2.40$) is of the same order as the distance from the initial guess to the optimal parameters of the best run ($d_1 = 2.04$). This is a general pattern: comparable (low) Errors can be obtained with significantly different parameter vectors. This is most likely due to the presence of several local minima with a similar depth in the Error function. To shed some light on the origin of distinct solutions with nearly the same Error, Figure 2 depicts the Error as function of a linear interpolation (in 1000 steps) between the solution from the best and second-best run (CMA-ES and educated guess), for the three training sets. These scans illustrate that the Error is not a convex function and may thus have several local minima. Our findings do not exclude the possibility that the Error is insensitive to certain linear combinations of parameters, which could also result in multiple solutions with a similar Error.[38]

The curves in Figure 2 also exhibit a significant degree of noise for the Disulfide and to larger extent for the Silica training set, in line with previous works.[32,40] To illustrate the severity of the noise, the Error of neighboring points in Figure 2b can differ by 1000 units, while the parameters change by less than 0.5%. Such levels of noise alone can create many local minima, most of which are irrelevant. This also explains why a recomputation of the Error for the Silica and Disulfide training sets in Table 1, using force-field parameters from the literature, can differ strongly from earlier publications. For these training sets, the Error itself is not robust, i.e. small changes parameters may have a large impact on the Error.

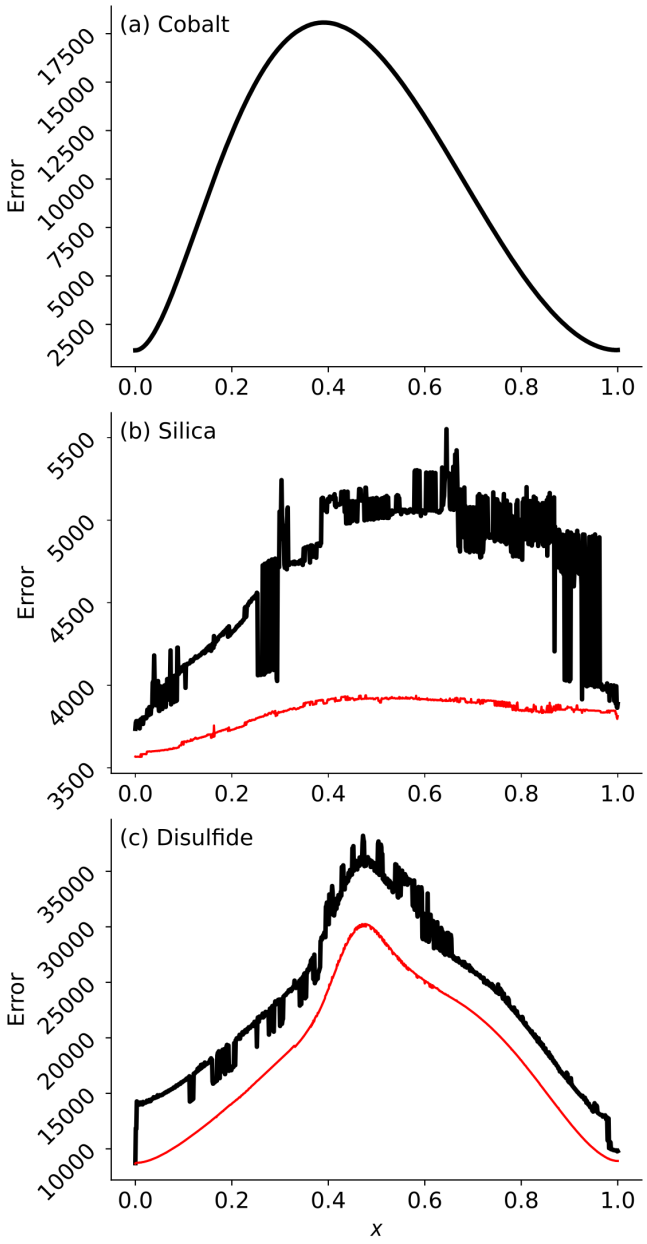

Figure 2: For the three training sets, a linear scan through parameter space is performed: $p_i^a + x(p_i^b - p_i^a)$, where $x \in [0, 1]$. The two end points of the scan, p_i^a and p_i^b, are the solutions of the best and second best run when optimizing the parameters with CMA-ES using the educated initial guess. At 1000 equidistant grid points for x, the total Error is computed and shown in the plots as a black curve. In case of the Silica and Disulfide training sets, the red curve is the Error without those contributions that cause large discontinuous jumps in the Error along the linear scan.

Because this noise in the Error degrades the performance of any optimization algorithm, the remainder of this section addresses its origins and explores mitigation strategies for future work.

A detailed analysis revealed that the jumps in Figure 2 are caused by 8 out of 309 items in the Silica training set and 11 out of 4875 items in the Disulfide training set. The Error without these noisy items (red curve) is much smoother. We repeated some of the CMA-ES runs after removing the problematic items from these training sets. While the lowest Error for the worst run decreases notably, the results for the best run do not improve significantly. This means that the problematic items in the training set mainly increase the risk that CMA-ES converges to a higher local minimum. In other words, one can also improve the robustness of a parameter optimization by designing training sets without noisy Error contributions.

To understand the origin of the noise in Figure 2, we investigated every term in the Error function, see Eq. (3), along the linear scan. Noise in a ReaxFF prediction, Δx_i, relative to a mean value $\langle x_i \rangle$, adds a contribution to the noise in the Error (to first order) comprising two factors:

$$\Delta\text{Error}_i \approx \frac{2}{\sigma_i^2} \underbrace{|\langle x_i \rangle - x_{i,\text{ref}}|}_{\text{factor 1}} \underbrace{\Delta x_i}_{\text{factor 2}} \tag{8}$$

When we observe significant noise in the Error, this can be either due to a large first or second factor. This is consistent with earlier work of Larsson et al,[40] where it was observed that the Error function becomes smoother near the optimal parameters, which can be explained by a decrease of the first factor.

For the Silica training set, apparently only 8 out of the 309 Error terms are responsible for the largest jumps in Figure 2. Several smaller discontinuities are also present for exactly the same reasons as the larger jumps. Most of the problematic terms in the Error are related to molecules for which at least two (but often many more) metastable conformations or configurations exist. Two examples are shown in Figure 3. The 12-membered silica ring in Figure 3a can have many slightly different conformations due to the high flexibility of the Si-

O-Si angles. Figure 3b represents the products of a silica condensation reaction. The water molecule is weakly bound to the condensed silica cluster, with two possible configurations differing in energy by 20 kcal mol^{-1}. The end result of the geometry optimization (either of the two states) depends erratically on the force field parameters, resulting in sudden changes of the Error by approximately 300 units with only tiny changes in parameters. One could reduce this sensitivity by (a) using more rigid molecules and (b) by including reaction products separately in the training set instead of combining them into a single complex. Another problematic case is the energy of a slightly expanded unit cell of quartz. The training set specifies that this geometry should only be optimized for five steps, instead of the usual 3000, without reaching convergence. Due to the large remaining atomic forces, small changes in geometry cause large differences in energy. The exact configuration after five steps depends unpredictably on force field parameters and algorithmic details of the geometry optimizer, which is another source of noise. In this case, allowing for more geometry iterations should resolve the issue, at the expense of an increased computational cost.

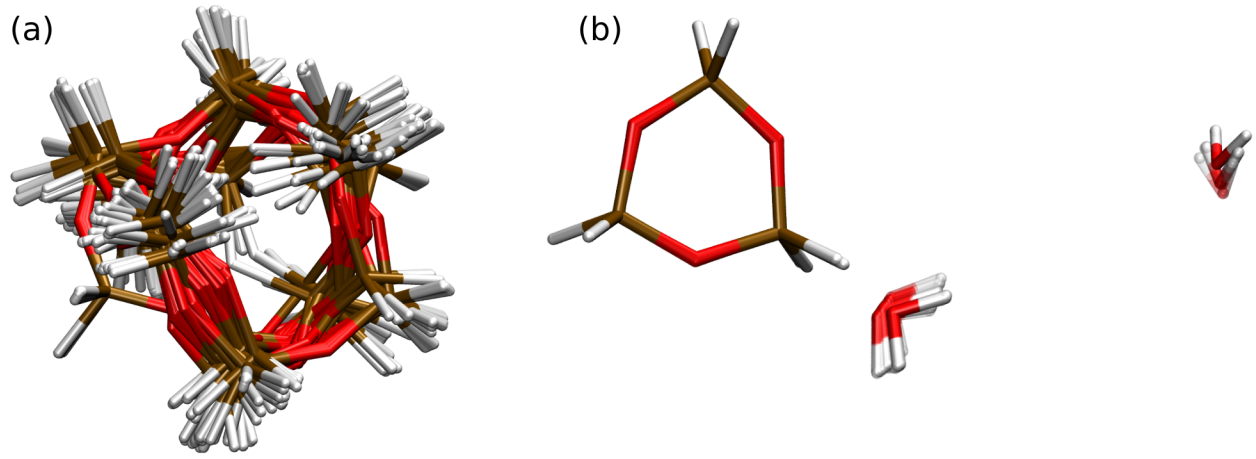

Figure 3: Two of the problematic molecules in the Silica training set responsible for discontinuous jumps in Figure 2b: (a) a 12-membered silica ring and (b) the product of a ring-closing condensation reaction of a linear silica trimer. For both molecules, 100 optimized geometries are shown in overlay, obtained with different ReaxFF parameters along the scan. In part (b), the water molecule, which is a product from this condensation reaction, is part of this geometry, for which ReaxFF predicts roughly two stable positions relative to the three-ring.

For the Disulfide training set, 11 out of 4875 Error terms are responsible for all the visible

noise in Figure 2. These 11 terms measure errors on dihedral angles (torsions about C-O and C-S bonds) in the four molecules shown in 4. The geometry optimization of the four molecules is usually not complete after 3000 steps. In case of convergence failure, the last geometry is used as the best available approximation of a converged result. However, due to the incomplete convergence, the internal coordinates contain a virtually random component, which is yet another source of noise.

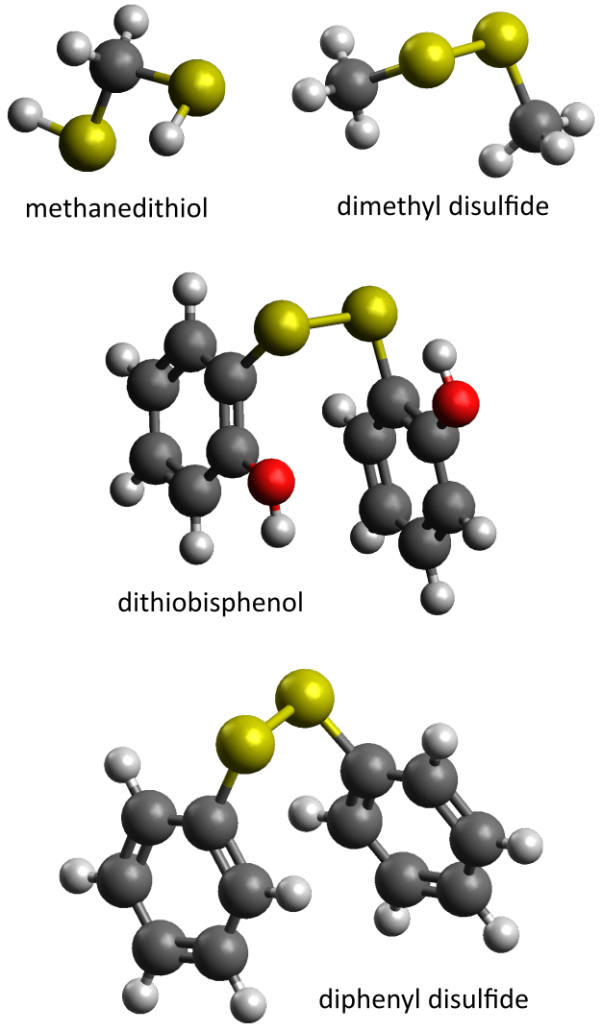

Figure 4: Four molecules in the Disulfide training set, whose Errors on the dihedral angles are responsible for the noise in Figure 2c.

Remarkably, there is no visible noise in Figure 2 for the Cobalt training set, for which there could be two explanations. First, the Cobalt training set only contains energy dif-

ferences, which are not sensitive to small deviations in the geometry, because the nuclear forces are nearly zero after the geometry optimization. A second possible explanation is that the Cobalt Reactive force field uses exclusively two-body terms, thereby eliminating some sources of noise present in three- and four-body energy terms.

The relation between geometry convergence and noise in the Error provides a second explanation for the observation of Larsson *et al.* that the Error becomes smoother near the optimal parameters. The geometries provided in the training set are normally the optimal ones that ReaxFF should reproduce. Hence, with good ReaxFF parameters, fewer geometry optimization steps are needed, resulting in a lower risk for geometry convergence failures and corresponding discontinuities in the Error function.

4 Conclusions and outlook

Our systematic comparison of the MCFF, CMA-ES and GA optimizers reveals that all three methods require multiple independent optimization runs to obtain parameters with a low Error. Particularly, a single optimization run with any of these methods, with reasonable default settings and an educated (or random) initial guess, has a significant risk of obtaining parameters with relatively high errors. The main reason is that the end result of these stochastic algorithms is affected in an unpredictable manner by the random seed and the initial guess.

Different optimization algorithms have their strengths and weaknesses. For example, CMA-ES is capable of finding the lowest Error for two out of three training sets, but does not find them systematically. Depending on the initial guess or the random seed, it may also converge to a higher local minimum. In contrast, GA is more robust, i.e. it exhibits the minimal difference between the lowest Error for independent runs.

With any of the tested optimizers, further tuning of algorithm settings may improve their performance and result in effective methods to optimize ReaxFF parameters. For example,

one can switch between different algorithms in subsequent runs to combine the robustness of GA with the CMA-ES local optimization efficiency. Alternatively, many independent short runs, e.g. with CMA-ES, can compensate for its risk of getting trapped in high local minima. Such advanced schemes will certainly be more effective than a single run with a single method, yet they also exhibit a larger number of hyperparameters, which may require case-by-case tuning.

For all three training sets in this work, independent optimization runs result in clearly different ReaxFF parameters with almost equally low Error values, which seems troublesome at first glance but in fact provides useful information. Intuitively, one may simply select the lowest minimum. However, different solutions of comparable quality can be used more effectively, which will be explored in future work. For example, one could also select a local minimum affected less by overfitting, with a low Error on a test set instead of the training set, in analogy to to early stopping. Another use case is improving the reliability of ReaxFF simulations with reinforcement learning. The spread on outcomes from production runs using different near-optimal parameters is a lower bound for the uncertainty on the prediction of interest. To reduce this uncertainty, we suggest to add properties of molecules to the training set, for which ReaxFF results vary with different near-optimal parameters. Including such reference data will narrow down the region in the parameter space where the Error is low, potentially reducing overfitting artifacts. This method of enhancing training sets should allow one to start even from an incomplete training set and continuously extend it until consistent predictions for a production run are obtained.

Our assessment also highlighted the importance of a robust geometry optimization for the calibration of ReaxFF parameters. When geometries converge poorly, the final ones contain a random component, which propagates to the Error and impairs the parameter calibration. In the course of this work, we have refined the geometry optimization algorithm used in the ADF2018 implementation of ReaxFF, resulting in a smoother Error function. While such improvements are clearly beneficial, discontinuities in the Error may still appear when

optimized geometries have a high sensitivity to the force field parameters, e.g. in case of very flexible systems in the training set with different possible conformations. In future work, convergence issues or multiple (meta)stable configurations could be detected automatically through a high sensitivity of the geometries to small changes in ReaxFF parameters.

Acknowledgement

We are indebted to M. Dittner, J. Müller and B. Hartke for providing a copy of the development version of OGOLEM and input files. This project received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation: Marie Skłodowska-Curie actions under grant agreement No. 641887 (DEFNET), Innovation in SMEs under grant agreement No. 739746 (NET) and Secure, clean and efficient energy under grant agreements No. 764810 (S4CE). T.V. acknowledges the Research Board of Ghent University (BOF) for its financial support. The computational resources and services used in this work were partially provided by the VSC (Flemish Supercomputer Center), funded by the FWO (Research Foundation Flanders).

Supporting Information Available

The changes to the literature values of the Silica parameters and the allowed intervals for these parameters during the calibration, a detailed description of the Torsion2013 correction, a derivation of initial and final inverse temperatures for MCFF, OGOLEM input files and plots of Error versus iteration for all parameter calibration reported in tables 3, 4 and 5 are documented in the Supporting Information. Also a ZIP file is provided with the (updated) training sets, initial and the optimal parameters from each calibration run.

References

(1) Hine, N. D. M.; Robinson, M.; Haynes, P. D.; Skylaris, C.-K.; Payne, M. C.; Mostofi, A. A. Accurate ionic forces and geometry optimization in linear-scaling density-functional theory with local orbitals. *Phys. Rev. B* **2011**, *83*, 195102.

(2) VandeVondele, J.; Borštnik, U.; Hutter, J. Linear Scaling Self-Consistent Field Calculations with Millions of Atoms in the Condensed Phase. *J. Chem. Theory Comput.* **2012**, *8*, 3565–3573.

(3) Schütt, O.; Messmer, P.; Hutter, J.; VandeVondele, J. In *Electronic Structure Calculations on Graphics Processing Units*; Walker, R. C., Götz, A. W., Eds.; John Wiley & Sons, Ltd, 2016; pp 173–190.

(4) van Schoot, H.; Visscher, L. In *Electronic Structure Calculations on Graphics Processing Units*; Walker, R. C., Götz, A. W., Eds.; John Wiley & Sons, Ltd, 2016; pp 101–114.

(5) Nishimura, Y.; Nakai, H. Parallel implementation of efficient charge-charge interaction evaluation scheme in periodic divide-and-conquer density-functional tight-binding calculations. *J. Comput. Chem.* **2018**, *39*, 105–116.

(6) Mohr, S.; Eixarch, M.; Amsler, M.; Mantsinen, M. J.; Genovese, L. Linear scaling DFT calculations for large tungsten systems using an optimized local basis. *Nucl. Mater. Energy* **2018**, *15*, 64–70.

(7) Daw, M. S.; Baskes, M. I. Semiempirical, quantum mechanical calculation of hydrogen embrittlement in metals. *Phys. Rev. Lett.* **1983**, *50*, 1285–1288.

(8) Tersoff, J. New empirical approach for the structure and energy of covalent systems. *Phys. Rev. B* **1988**, *37*, 6991–7000.

(9) Brenner, D. W.; Shenderova, O. A.; Harrison, J. A.; Stuart, S. J.; Ni, B.; Sinnott, S. B.

A second-generation reactive empirical bond order (REBO) potential energy expression for hydrocarbons. *J. Phys. Condens. Mat.* **2002**, *14*, 783–802.

(10) Phillpot, S. R.; Antony, A. C.; Shi, L.; Fullarton, M. L.; Liang, T.; Sinnott, S. B.; Zhang, Y.; Biner, S. B. Charge Optimized Many Body (COMB) potentials for simulation of nuclear fuel and clad. *Comput. Mater. Sci.* **2018**, *148*, 231–241.

(11) van Duin, A. C. T.; Dasgupta, S.; Lorant, F.; Goddard, W. A. ReaxFF: A Reactive Force Field for Hydrocarbons. *J. Phys. Chem. A* **2001**, *105*, 9396–9409.

(12) Liang, T.; Shin, Y. K.; Cheng, Y.-T.; Yilmaz, D. E.; Vishnu, K. G.; Verners, O.; Zou, C.; Phillpot, S. R.; Sinnott, S. B.; van Duin, A. C. Reactive Potentials for Advanced Atomistic Simulations. *Annu. Rev. Mater. Res.* **2013**, *43*, 109–129.

(13) Han, Y.; Jiang, D.; Zhang, J.; Li, W.; Gan, Z.; Gu, J. Development, applications and challenges of ReaxFF reactive force field in molecular simulations. *Front. Chem. Sci. Eng.* **2016**, *10*, 16–38.

(14) Senftle, T. P.; Hong, S.; Islam, M. M.; Kylasa, S. B.; Zheng, Y.; Shin, Y. K.; Junkermeier, C.; Engel-Herbert, R.; Janik, M. J.; Aktulga, H. M.; Verstraelen, T.; Grama, A.; van Duin, A. C. T. The ReaxFF reactive force-field: development, applications and future directions. *npj Comput. Mater.* **2016**, *2*, 15011.

(15) Qian, H.-J.; van Duin, A. C. T.; Morokuma, K.; Irle, S. Reactive Molecular Dynamics Simulation of Fullerene Combustion Synthesis: ReaxFF vs DFTB Potentials. *J. Chem. Theory Comput.* **2011**, *7*, 2040–2048.

(16) Boes, J. R.; Groenenboom, M. C.; Keith, J. A.; Kitchin, J. R. Neural network and ReaxFF comparison for Au properties. *Int. J. Quant. Chem.* **2016**, *116*, 979–987.

(17) Chenoweth, K.; van Duin, A. C. T.; Goddard, W. A. ReaxFF Reactive Force Field for

Molecular Dynamics Simulations of Hydrocarbon Oxidation. *J. Phys. Chem. A* **2008**, *112*, 1040–1053.

(18) Ashraf, C.; van Duin, A. C. Extension of the ReaxFF Combustion Force Field toward Syngas Combustion and Initial Oxidation Kinetics. *J. Phys. Chem. A* **2017**, *121*, 1051–1068.

(19) Berman, D.; Deshmukh, S. A.; Sankaranarayanan, S. K. R. S.; Erdemir, A.; Sumant, A. V. Extraordinary Macroscale Wear Resistance of One Atom Thick Graphene Layer. *Adv. Funct. Mater.* **2014**, *24*, 6640–6646.

(20) Liu, L.; Liu, Y.; Zybin, S. V.; Sun, H.; Goddard, W. A. ReaxFF-lg: Correction of the ReaxFF Reactive Force Field for London Dispersion, with Applications to the Equations of State for Energetic Materials. *J. Phys. Chem. A* **2011**, *115*, 11016–11022.

(21) Nielson, K. D.; van Duin, A. C. T.; Oxgaard, J.; Deng, W.-Q.; Goddard, W. A. Development of the ReaxFF Reactive Force Field for Describing Transition Metal Catalyzed Reactions, with Application to the Initial Stages of the Catalytic Formation of Carbon Nanotubes. *J. Phys. Chem. A* **2005**, *109*, 493–499.

(22) Zou, C.; van Duin, A. C. T. Investigation of Complex Iron Surface Catalytic Chemistry Using the ReaxFF Reactive Force Field Method. *JOM* **2012**, *64*, 1426–1437.

(23) Islam, M. M.; Kolesov, G.; Verstraelen, T.; Kaxiras, E.; van Duin, A. C. T. eReaxFF: A Pseudoclassical Treatment of Explicit Electrons within Reactive Force Field Simulations. *J. Chem. Theory Comput.* **2016**, *12*, 3463–3472.

(24) Budzien, J.; Thompson, A. P.; Zybin, S. V. Reactive Molecular Dynamics Simulations of Shock Through a Single Crystal of Pentaerythritol Tetranitrate. *J. Phys. Chem. B* **2009**, *113*, 13142–13151.

(25) Gale, J. D.; Raiteri, P.; van Duin, A. C. T. A reactive force field for aqueous-calcium carbonate systems. *Phys. Chem. Chem. Phys.* **2011**, *13*, 16666.

(26) Nakano, A.; Kalia, R. K.; Nomura, K.-i.; Sharma, A.; Vashishta, P.; Shimojo, F.; van Duin, A. C. T.; Goddard, W. A.; Biswas, R.; Srivastava, D. A divide-and-conquer/cellular-decomposition framework for million-to-billion atom simulations of chemical reactions. *Comput. Mat. Sci.* **2007**, *38*, 642–652.

(27) Aktulga, H.; Fogarty, J.; Pandit, S.; Grama, A. Parallel reactive molecular dynamics: Numerical methods and algorithmic techniques. *Parallel Comput.* **2012**, *38*, 245–259.

(28) Kamat, A. M.; van Duin, A. C. T.; Yakovlev, A. Molecular Dynamics Simulations of Laser-Induced Incandescence of Soot Using an Extended ReaxFF Reactive Force Field. *J. Phys. Chem. A* **2010**, *114*, 12561–12572.

(29) van Duin, A. C. T.; Strachan, A.; Stewman, S.; Zhang, Q.; Xu, X.; Goddard, W. A. ReaxFF_SiO Reactive Force Field for Silicon and Silicon Oxide Systems. *J. Phys. Chem. A* **2003**, *107*, 3803–3811.

(30) Fogarty, J. C.; Aktulga, H. M.; Grama, A. Y.; van Duin, A. C. T.; Pandit, S. A. A reactive molecular dynamics simulation of the silica-water interface. *J. Chem. Phys.* **2010**, *132*, 174704.

(31) LaBrosse, M. R.; Johnson, J. K.; van Duin, A. C. T. Development of a Transferable Reactive Force Field for Cobalt. *J. Phys. Chem. A* **2010**, *114*, 5855–5861.

(32) Iype, E.; Hütter, M.; Jansen, A. P. J.; Nedea, S. V.; Rindt, C. C. M. Parameterization of a reactive force field using a Monte Carlo algorithm. *J. Comput. Chem.* **2013**, *34*, 1143–1154.

(33) van Duin, A. C. T.; Zou, C.; Joshi, K.; Bryantsev, V.; Goddard, W. A. In *Computational*

Catalysis; Asthagiri, A., Janik, M. J., Eds.; The Royal Society of Chemistry, 2013; Chapter 6, pp 223–243.

(34) Müller, J.; Hartke, B. Reax FF Reactive Force Field for Disulfide Mechanochemistry, Fitted to Multireference ab Initio Data. *J. Chem. Theory Comput.* **2016**, *12*, 3913–3925.

(35) Barcaro, G.; Monti, S.; Sementa, L.; Carravetta, V. Parametrization of a Reactive Force Field (ReaxFF) for Molecular Dynamics Simulations of Si Nanoparticles. *J. Chem. Theory Comput.* **2017**, *13*, 3854–3861.

(36) van Duin, A. C. T.; Baas, J. M. A.; van de Graaf, B. Delft molecular mechanics: a new approach to hydrocarbon force fields. Inclusion of a geometry-dependent charge calculation. *J. Chem. Soc., Faraday Trans.* **1994**, *90*, 2881–2895.

(37) Dittner, M.; Müller, J.; Aktulga, H. M.; Hartke, B. Efficient global optimization of reactive force-field parameters. *J. Comput. Chem.* **2015**, *36*, 1550–1561.

(38) Pahari, P.; Chaturvedi, S. Determination of best-fit potential parameters for a reactive force field using a genetic algorithm. *J. Mol. Mod.* **2012**, *18*, 1049–1061.

(39) Jaramillo-Botero, A.; Naserifar, S.; Goddard, W. A. General Multiobjective Force Field Optimization Framework, with Application to Reactive Force Fields for Silicon Carbide. *J. Chem. Theory Comput.* **2014**, *10*, 1426–1439.

(40) Larsson, H. R.; van Duin, A. C. T.; Hartke, B. Global optimization of parameters in the reactive force field ReaxFF for SiOH. *J. Comput. Chem.* **2013**, *34*, 2178–2189.

(41) Larentzos, J. P.; Rice, B. M.; Byrd, E. F. C.; Weingarten, N. S.; Lill, J. V. Parameterizing Complex Reactive Force Fields Using Multiple Objective Evolutionary Strategies (MOES). Part 1: ReaxFF Models for Cyclotrimethylene Trinitramine (RDX) and 1,1-Diamino-2,2-dinitroethene (FOX-7). *J. Chem. Theory Comput.* **2015**, *11*, 381–391.

(42) Deetz, J. D.; Faller, R. Parallel Optimization of a Reactive Force Field for Polycondensation of Alkoxysilanes. *J. Phys. Chem. B* **2014**, *118*, 10966–10978.

(43) Hu, X.; Schuster, J.; Schulz, S. E. Multiparameter and Parallel Optimization of ReaxFF Reactive Force Field for Modeling the Atomic Layer Deposition of Copper. *J. Phys. Chem. C* **2017**, *121*, 28077–28089.

(44) Kirkpatrick, S.; Gelatt, C. D.; Vecchi, M. P. Optimization by Simulated Annealing. *Science* **1983**, *220*, 671–680.

(45) Hansen, N. In *Towards a New Evolutionary Computation. Advances on Estimation of Distribution Algorithms*; Lozano, J. A., Larrañaga, P., Inza, I., Bengoetxea, E., Eds.; Studies in Fuzziness and Soft Computing; Springer Berlin Heidelberg, 2006; Vol. 192; pp 75–102.

(46) Trnka, T.; Tvaroška, I.; Koča, J. Automated Training of ReaxFF Reactive Force Fields for Energetics of Enzymatic Reactions. *J. Chem. Theory Comput.* **2018**, *14*, 291–302.

(47) Barnes, B. C.; Gelb, L. D. Meta-Optimization of Evolutionary Strategies for Empirical Potential Development: Application to Aqueous Silicate Systems. *J. Chem. Theory Comput.* **2007**, *3*, 1749–1764.

(48) Tafipolsky, M.; Schmid, R. Systematic First Principles Parameterization of Force Fields for Metal-Organic Frameworks using a Genetic Algorithm Approach. *J. Phys. Chem. B* **2009**, *113*, 1341–1352.

(49) Ivanov, M. V.; Talipov, M. R.; Timerghazin, Q. K. Genetic Algorithm Optimization of Point Charges in Force Field Development: Challenges and Insights. *J. Phys. Chem. A* **2015**, *119*, 1422–1434.

(50) France-Lanord, A.; Soukiassian, P.; Glattli, C.; Wimmer, E. Ab initio parameteriza-

tion of a charge optimized many-body forcefield for Si–SiO_2: Validation and thermal transport in nanostructures. *J. Chem. Phys.* **2016**, *144*, 104705.

(51) Rodríguez-Fernández, R.; Pereira, F. B.; Marques, J. M.; Martínez-Núñez, E.; Vázquez, S. A. GAFit: A general-purpose, user-friendly program for fitting potential energy surfaces. *Comput. Phys. Commun.* **2017**, *217*, 89–98.

(52) Zahariev, F.; De Silva, N.; Gordon, M. S.; Windus, T. L.; Dick-Perez, M. ParFit: A Python-Based Object-Oriented Program for Fitting Molecular Mechanics Parameters to ab Initio Data. *J. Chem. Inf. Model.* **2017**, *57*, 391–396.

(53) Dürholt, J. P.; Fraux, G.; Coudert, F.-X.; Schmid, R. Ab Initio Derived Force Fields for Zeolitic Imidazolate Frameworks: MOF-FF for ZIFs. *J. Chem. Theory Comput.* **2019**, *15*, 2420–2432.

(54) Moqadam, M.; Riccardi, E.; Trinh, T. T.; Åstrand, P.-O.; van Erp, T. S. A test on reactive force fields for the study of silica dimerization reactions. *J. Chem. Phys.* **2015**, *143*, 184113.

(55) Hubin, P. O.; Jacquemin, D.; Leherte, L.; Vercauteren, D. P. Parameterization of the ReaxFF reactive force field for a proline-catalyzed aldol reaction. *J. Comput. Chem.* **2016**, *37*, 2564–2572.

(56) Database of ReaxFF parameters managed by SCM. `https://www.scm.com/doc/ReaxFF/Included_Forcefields.html`, Accessed May 24, 2018.

(57) Hansen, N.; Kern, S. In *Parallel Problem Solving from Nature - PPSN VIII*; Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G., Yao, X., Burke, E. K., Lozano, J. A., Smith, J., Merelo-Guervós, J. J., Bullinaria, J. A., Rowe, J. E., Tiňo, P., Kabán, A., Schwefel, H.-P., Eds.; Springer Berlin Heidelberg: Berlin, Heidelberg, 2004; Vol. 3242; pp 282–291.

(58) Verstraelen, T.; Bultinck, P. Can the electronegativity equalization method predict spectroscopic properties? *Spectrochim. Acta A* **2013**, *136 Pt A*, 76–80.

(59) Semino, R.; Dürholt, J. P.; Schmid, R.; Maurin, G. Multiscale Modeling of the HKUST-1/Poly(vinyl alcohol) Interface: From an Atomistic to a Coarse Graining Approach. *J. Phys. Chem. C* **2017**, *121*, 21491–21496.

(60) Hansen, N. CMA-ES written in ANSI C. `https://github.com/CMA-ES/c-cmaes`, Accessed February 9, 2018.

Graphical TOC Entry