JAX-ReaxFF: Gradient-Based Framework for Fast Optimization of Reactive Force Fields

Mehmet Cagri Kaymak, Ali Rahnamoun, Kurt A. O'Hearn, Adri C. T. van Duin*, Kenneth M. Merz, Jr., and Hasan Metin Aktulga*

NSF Project: CDS&E: ReaxFF2: Efficient and Scalable Methods for Long-time Reactive Molecular Dynamics Simulations (Award #1807622)
* Project PIs

Overview

- Force fields are sets of parameterized mathematical equations describing the intra- and inter-molecular interactions; their use is very common in molecular dynamics (MD).
- The reactive force field (ReaxFF) model bridges the gap between classical MD models and quantum mechanical (QM) models by incorporating dynamic bonding and polarizability; it has a very complex functional form with tens to hundreds of parameters to be optimized depending on the target application.
- Existing parameter optimization methods for ReaxFF consist of black-box techniques using genetic algorithms or Monte-Carlo methods; due to the stochastic nature of these methods, the optimization process typically requires hundreds of thousands to millions of error evaluations for complex tasks.
- **JAX-ReaxFF** is a novel software tool that leverages modern ML infrastructure to enable fast optimization of ReaxFF parameters .
- JAX-ReaxFF calculates gradients of the loss function using the auto-differentiation functionality in the JAX library, and utilizes highly effective local optimization methods initiated from multiple guesses to avoid being trapped in poor local minima.
- Leveraging the architectural portability of the JAX framework, JAX-ReaxFF can execute efficiently on multi-core CPUs, graphics processing units (GPUs), or even tensor processing units (TPUs).
- Overall, we are able to decrease ReaxFF parameter optimization time from days to mere minutes.

Introduction and Background

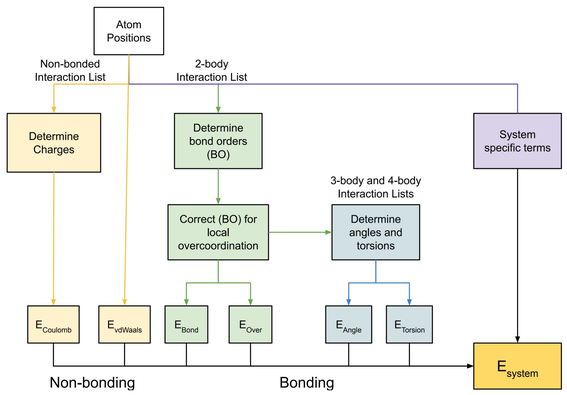


Figure 1: Flow graph of calculations performed in ReaxFF

Type	Training Item	Target	Description
Charge	ID1 1	0.5	Charge for atom 1 in the corresponding molecular structure (elementary units)
Energy	ID1/1 - ID2/2 - ID3/3	50	Scaled energy difference between specified geometries (kcal/mol)
	ID1/1	-150	
	ID3/2 - ID1/3	30	
Geometry	ID1 1 2	1.25	Distance between atoms 1 and 2 (°A)
	ID2 1 2 3	120	Valence angle between atoms 1, 2 and 3 (in degrees)
	ID3 1 2 3 4	170	Torsion angle between atoms 1, 2, 3 and 4 (in degrees)
Force	ID1 1	0.5 0.5 0.5	Forces on atom 1 (kcal/mol °A)
	ID2	1.0	Root mean squared grads. (kcal/mol °A)

Table 1: Commonly used training item types in ReaxFF training. Identifiers (e.g., ID1, ID2 and ID3) denote geometries, i.e., atomic structures or molecules.

$$\mathrm{Error}(m) = \sum_{i=1}^{N} \left(\frac{x_i - y_i}{\sigma_i} \right)^2$$

Different training items are combined into a loss (fitness) function, where m is the model, x_i is the prediction, y_i is the ground truth as given in the reference dataset, and σ^{-1}_i is the weight assigned to each training item.

Proposed Method

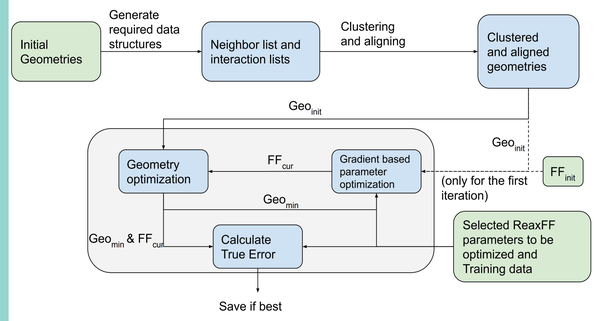


Figure 2: JAX-ReaxFF execution flow graph.

Clustering and alignment for SIMD parallelization

- To speedup the computation, we utilize just-in-time compilation via XLA to create a static computational graph and map it to the desired computational device. JAX-JIT requires array sizes to be known and arrays to be properly aligned.
- A modified version of K-Means algorithm (with a unique distance metric) is designed to automatically cluster and align computationally similar geometries together to minimize the padding.
- The distance between geometry g_i and cluster C_j is given as:

$$\mathrm{Dist}\,(C_j, g_i) = \mathrm{Cost}\,(C_j \cup \{g_i\}) - \mathrm{Cost}\,(C_j)$$

$$\mathrm{Cost}\,(C) = s \cdot [w_1 n_1 + w_2 n_2 + w_3 n_3 + w_4 n_4 + w_5 n_5 n_1^2]$$

n_1, n_2, n_3, n_4, and n_5 are the max numbers of atoms, 2-body interactions, 3-body interactions, 4-body interactions, and periodic boxes within cluster C, respectively. w_1 through w_5 are indicators of the relative computational costs.

Gradient-based local optimization

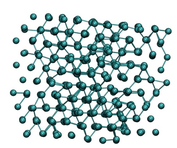
Algorithm 1 Gradient-Based Local Optimization

$\mathrm{FF}_{\mathrm{cur}} \leftarrow \mathrm{FF}_{\mathrm{init}}$
for $iteration = 1, 2, \ldots$ **do**
  $\mathrm{FF}_{\mathrm{cur}} \leftarrow$ Locally minimize the error through the selected gradient-based algorithm using $\mathrm{Geo}_{\mathrm{min}}$ and starting from $\mathrm{FF}_{\mathrm{cur}}$. $\mathrm{Geo}_{\mathrm{min}}$ is fixed.
  $\mathrm{Geo}_{\mathrm{min}} \leftarrow$ Geometry optimize the structures starting from the initial geometries $\mathrm{Geo}_{\mathrm{init}}$ with the current model $\mathrm{FF}_{\mathrm{cur}}$
  $E_{\mathrm{cur}} \leftarrow$ Calculate the current error using $\mathrm{Geo}_{\mathrm{min}}$ and $\mathrm{FF}_{\mathrm{cur}}$
  if $E_{\mathrm{cur}} < E_{\mathrm{best}}$ **then**
    $E_{\mathrm{best}} \leftarrow E_{\mathrm{cur}}$
    $\mathrm{FF}_{\mathrm{best}} \leftarrow \mathrm{FF}_{\mathrm{cur}}$
  end if
  if $|E_{\mathrm{cur}} - E_{\mathrm{prev}}| / E_{\mathrm{prev}} < 0.001$ **then**
    $FF_{\mathrm{cur}} \leftarrow$ Add small uniform noise to FF_{best}
  end if
  $E_{\mathrm{prev}} \leftarrow E_{\mathrm{cur}}$
end for

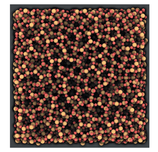
- Some geometries might require energy minimization. JAX needs to trace the energy minimization routine to calculate the gradients which is error-prone due to the complexity.
- To remedy this problem, we separate the geometry optimization and use two different error calculations. The first one calculates the error without energy minimization (surrogate error) and the second one calculates the true error with energy minimization.
- The surrogate error helps us easily calculate the gradients and accelerate the training.

Evaluation and Results

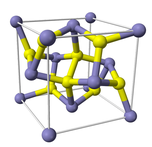
A mix of materials and molecules with different properties and various types of items (single point energies, partial charges, geometry optimization, force matching) in their training datasets have been used for evaluation.



Cobalt (Co)



Amorphous Silica (SiO_2)



FeS_2 (Fool's Gold) [1]

Training Dataset	Num. Params	Num. Geos	Num. Minim.	Charge Data	Geo Data	Force Data	Energy Data
Cobalt	12	146	130	0	0	0	144
Silica	67	302	221	5	26	0	265
Disulfide	87	231	10	0	255	4401	219

Table 2: Training data sets

JAX-ReaxFF was compared against previously published methods, i.e. CMA-ES, MCFF and GA that were reported in [2], using two different local optimization methods: LBFGSB and SLSQP. Comparison criteria include the number of loss function evalautions and average execution time. Only JAX-ReaxFF can benefit from GPU acceleration, which is automatically enabled by JAX.

Method	Lowest Error	Median Error	Avg. # Single Step Eval.	# True Eval.	Avg. CPU Time (min)	Avg. GPU Time (min)
LBFGSB	1368	2334	480	20	23.5	1.2
SLSQP	1191	2253	513	20	24.8	1.3
Gen. Alg.	1346	1645	-	200k	3913	-
CMA-ES	1150	1894	-	45k	880	-
MCFF	1422	2104	-	45k	880	-

Table 3: Cobalt training results

Method	Lowest Error	Median Error	Avg. # Single Step Eval.	# True Eval.	Avg. CPU Time (min)	Avg. GPU Time (min)
LBFGSB	3901	5214	1865	20	25.0	1.6
SLSQP	3870	4498	2962	20	31.9	2.0
Gen. Alg.	3577	3748	-	200k	1632	-
CMA-ES	3749	4753	-	45k	367	-
MCFF	5059	6584	-	45k	367	-

Table 4: Silica training results

Method	Lowest Error	Median Error	Avg. # Single Step Eval.	# True Eval.	Avg. CPU Time (min)	Avg. GPU Time (min)
LBFGSB	6513	12906	2017	20	13.5	1.2
SLSQP	4744	43712	1059	20	22.2	2.0
Gen. Alg.	19285	20384	-	200k	1632	-
CMA-ES	8052	11371	-	45k	367	-
MCFF	8507	11893	-	45k	367	-

Table 5: Disulfide training results

- These results show that the training errors of the force fields optimized using JAX-ReaxFF are on par with or better than those from the literature, while the training time decreases by one to three orders of magnitude.
- The relatively good median training errors obtained from multiple runs also indicate that JAX-ReaxFF could produce various force fields with comparable performance.
- The main benefit is that the overall training times are significantly lower which enables researchers to quickly iterate over ideas, try different weighting of the training items, or use different data for their target applications.

Validation

Force field optimization ensures a good fit to the loss function, but this does not guarantee high fidelity simulations in practice. To validate the fitted cobalt, silica and disulphide ReaxFF parameters, the optimized force fields were evaluated in actual MD simulations.

Property	2010 Force field	2019 Force field	JAX-ReaxFF Force field
Density	2.19	2.31	2.23
Si Coordination	3.99	3.94	3.97
O Coordination	1.99	1.97	1.99

Table 6: Silica properties calculated using three different force fields. The experimental value for silica density is 2.2 g/cm^3. Silicon and Oxygen coordination numbers are 4 and 2, respectively.

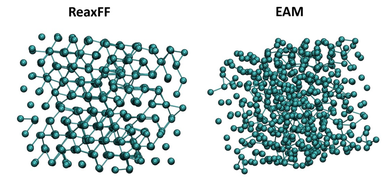


Figure 3: Final configurations of pure Co fcc crystals after annealing loop with 1000K-3000K temperature range. ReaxFF simulations performed using the new JAX-ReaxFF force field can preserve the crystal structure; this is the expected behavior. However, in simulations with the EAM force field, which is a popular model for metals, the crystal structure is heavily distorted.

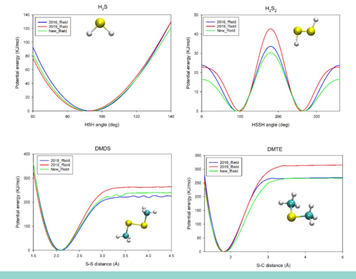


Figure 4: Potential energy graphs of energy minimized molecules including Sulfur bonds with dfferent restraints, calculated with the new JAX-ReaxFF force field and previously published force fields closely match each other.

Conclusions

- JAX-ReaxFF enables fast optimization of Reax force field parameters by leveraging the computational infrastructure advances in machine learning.
- JAX-ReaxFF uses several innovative techniques for high performance on architectures with GPUs. Clustering similar geometries together to maximize the SIMD parallelism while limiting the padding for alignment yields high parallelism, especially for single step evaluations.
- As described in Algorithm 1, by using single step energy evaluation-based approximations to the error function and gradient information about the search space, we are able to decrease the convergence time significantly with the help of GPU acceleration.
- We have empirically showed that even though the local optimizer is not fully aware of the geometry optimization, the overall algorithm converges with minimal changes in the parameter space as the algorithm progresses.
- JAX-ReaxFF provides a utility not available in other competing tools – its auto-differentiation functionality enables the study of the new functional forms for the ReaxFF model without explicitly implementing the force calculations or the optimizer, since both forces and parameter gradients can automatically be calculated by JAX.

Acknowledgements and References

This research was supported in part an NSF CDS&E grant (award number 1807622) and an NIH grant (award number GM130641). This work used computational resources provided by the Institute for Cyber-Enabled Research at Michigan State University.

[1] 'Disulphide' (2022) Wikipedia. Available at: https://en.wikipedia.org/wiki/Disulfide (Accessed: 19 July 2022).
[2] Shchygol, G., Yakovlev, A., Trnka, T., Van Duin, A.C. and Verstraelen, T., (2019). ReaxFF parameter optimization with Monte-Carlo and evolutionary algorithms: guidelines and insights. Journal of Chemical Theory and Computation, 15(12), pp.6799-6812.