Extending OpenKIM with an Uncertainty Quantification Toolkit for Molecular Modeling

Yonatan Kurniawan
Department of Physics and Astronomy
Brigham Young University
Provo, UT, USA
yonatank@byu.edu, ORCID 0000-0002-5369-5710

Cody L. Petrie
Department of Physics and Astronomy
Brigham Young University
Provo, UT, USA
codypetrie89@gmail.com, ORCID 0000-0002-5596-8516

Mark K. Transtrum
Department of Physics and Astronomy
Brigham Young University
Provo, UT, USA
mktranstrum@byu.edu, ORCID 0000-0001-9529-9399

Ellad B. Tadmor
Department of Aerospace Engineering and Mechanics
University of Minnesota
Minneapolis, MN, USA
tadmor@umn.edu, ORCID 0000-0003-3311-6299

Ryan S. Elliott
Department of Aerospace Engineering and Mechanics
University of Minnesota
Minneapolis, MN, USA
relliott@umn.edu, ORCID 0000-0003-4988-8306

Daniel S. Karls
Department of Aerospace Engineering and Mechanics
University of Minnesota
Minneapolis, MN, USA
karl0100@umn.edu, ORCID 0000-0002-4069-396X

Mingjian Wen
Energy Technologies Area
Lawrence Berkeley National Laboratory
Berkeley, CA, USA
mjwen@lbl.gov, ORCID 0000-0003-0013-575X

***Abstract*—Atomistic simulations are an important tool in materials modeling. Interatomic potentials (IPs) are at the heart of such molecular models, and the accuracy of a model's predictions depends strongly on the choice of IP. Uncertainty quantification (UQ) is an emerging tool for assessing the reliability of atomistic simulations. The Open Knowledgebase of Interatomic Models (OpenKIM) is a cyberinfrastructure project whose goal is to collect and standardize the study of IPs to enable transparent, reproducible research. Part of the OpenKIM framework is the Python package, KIM-based Learning-Integrated Fitting Framework (KLIFF), that provides tools for fitting parameters in an IP to data. This paper introduces a UQ toolbox extension to KLIFF. We focus on two sources of uncertainty: variations in parameters and inadequacy of the functional form of the IP. Our implementation uses parallel-tempered Markov chain Monte Carlo (PTMCMC), adjusting the sampling temperature to estimate the uncertainty due to the functional form of the IP. We demonstrate on a Stillinger–Weber potential that makes predictions for the atomic energies and forces for silicon in a diamond configuration. Finally, we highlight some potential subtleties in applying and using these tools with recommendations for practitioners and IP developers.**

***Index Terms*—Interatomic potential, MCMC, uncertainty quantification, OpenKIM**

I. Introduction

Molecular modeling is an important part of materials science, and interatomic potentials (IPs) are at the heart of most molecular modeling simulations [1]. Given the large number of IPs constructed for various applications, there is an acute need to standardize their computational implementation and evaluation and facilitate portability among researchers. To this end, the Open Knowledgebase of Interatomic Models (OpenKIM) was founded to make atomistic-scale simulations reliable, reproducible, and accessible [2], [3]. While the OpenKIM framework provides many tools for materials simulations, it does not provide a standardized tool for uncertainty quantification (UQ). In this paper, we describe a novel UQ toolbox within the OpenKIM framework, targeted at molecular modeling.

The general goal of molecular modeling is to predict properties of materials by simulating collections of atoms. In this setting, IPs are used to approximate the interaction energy of atoms as functions of the atomic positions and species [4]. IPs are used in conjunction with a simulation program, such as ASE [5] or LAMMPS [6], to model atomic behavior and extract material properties. Such simulations can be static, e.g., minimizing energy to obtain the equilibrium lattice parameter of a crystal, or dynamic, e.g. applying the fluctuation–dissipation theorem to compute thermal conductivity. The accuracy of the material properties predicted by an atomistic simulation depends strongly on the choice of IP,

and considerable effort has gone into developing IPs that are accurate for specific applications.

UQ is an emerging field of applied mathematics that aims to quantify and reduce uncertainties in mathematical models [7]. In molecular modeling, UQ can help assess the reliability of conclusions drawn from atomistic simulations. It is widely recognized that the largest source of uncertainty in molecular modeling is the functional form of the IP. Additional uncertainty comes from the values of the corresponding parameters, which are typically fit to experimental data or first-principles (quantum mechanical) calculations [8]. Having been fit to data, these IPs are often used to calculate out-of-sample material properties or to conduct large scale simulations. These simulations generally suffer from inconsistent transferability, i.e., they may struggle to accurately predict material properties to which the IPs are not fit [9]. The UQ process is especially important for assessing the reliability of these out-of-sample predictions.

There are many UQ methods that have been used in molecular modeling, including F-statistics estimations [10], ANOVA-based methods [11], multi-objective optimization [12], and profile likelihoods [13]. Among these methods, the most frequently utilized is a Bayesian method known as Markov chain Monte Carlo (MCMC) [14]–[25]. As such, we focus on MCMC methods, though we anticipate future extensions that will make other tools available to the molecular modeling community.

A universal challenge in applying UQ methods to multi-parameter models, such as IPs, is that models are often *sloppy* [13], [26], [27]. Sloppiness refers to an extremely ill-conditioned inference problem when fitting parameters to data, which is ubiquitous in many scientific fields [28]–[37]. For sloppy models, many parameter combinations are not well-constrained by available data and dubbed *practically unidentifiable*. This poses a number of challenges to clearly formulating and interpreting UQ results, as shown, for example, for molecular modeling [13], [38]. These subtleties need to be considered, as we demonstrate later in this paper, when performing UQ analyses.

Although many software or libraries for performing Bayesian sampling or other UQ methods exist, such as emcee [39], Chaospy [40], and EasyVVUQ [41] to name a few, but there are only few libraries that integrate UQ for molecular modeling, such as *potfit* [25], [42]. In this paper we introduce a UQ toolkit integrated within the OpenKIM framework to facilitate the application of UQ to molecular modeling. Integrating UQ capabilities directly into the OpenKIM framework allows for more uniform, reproducible results and helps to standardize the practice of reporting uncertainty as part of a molecular modeling workflow. In Sec. II, we introduce the theory behind UQ and describe in more detail the OpenKIM framework. We describe our specific UQ implementation in Sec. III and how to use it, with an example, in Sec. IV. Finally, we conclude and describe future directions for this UQ toolkit in Sec. V.

II. Background

A. Theory of Uncertainty Quantification

Uncertainty in modeling may have many sources such as stochasticity in data or numerical discretization. In molecular modeling, the dominant source of uncertainty is due to the functional form of the IP, sometimes known as *model inadequacy*. That is to say, the IP does not capture all of the physics present in the process it is intended to mimic. An IP is meant to encompass some effects of quantum mechanics, but it necessarily does not include all quantum effects, leading to uncertainty when the IP predicts material properties different from those on which it was trained. In this section, we describe a basic theory for estimating uncertainty due to model inadequacy by *inflating* the uncertainty in the model's parameters. The basic idea is to introduce fluctuations in the model's parameters with a scale comparable to the accuracy of the model.

In this formulation, we assume a collection of data $\{y_m\}_{m=1}^M$ and a parameterized family of model predictions $\{f_m(\theta)\}_{m=1}^M$. In our notation, θ are the parameters of the model and M denotes the number of observed data. The data correspond to predictions that the model can make, which include primitive quantities such as energy, forces and stress, or more complex material properties such as equilibrium lattice constants or thermal conductivity. Data for these quantities can be obtained experimentally or from more accurate first-principles calculations, such as density functional theory (DFT). A commonly used IP fitting method, force-matching, uses the energy, forces, and stress for a set of atomic configurations with DFT calculations as data [8], [27], [43]. However, data for other material properties, such as lattice parameters [8] and thermal conductivity [21], [22], are also used.

To compare model predictions against the data, we introduce a *cost* (or loss) function. The most commonly used cost is (weighted) least squares,

$$C(\theta) = \frac{1}{2}\sum_{m=1}^{M} w_m(y_m - f_m(\theta))^2, \tag{1}$$

where w_m are the weights.

Selection of the weights is an important first step in quantifying uncertainty. Since potentials are often trained for material properties that carry different physical units, such as eV for energy and eV/Å for forces, data should be appropriately weighted to put them on a common scale. Functionally, the role of the weights is to quantify the relative target accuracy for each of the model's predictions. When random errors in the data are the dominant source of uncertainty, w_m are often taken to be the inverse square of the standard error (error bars) of the experiments. However, since the dominant source of error in molecular modeling is the functional form of the IP, rather than errors in the data, some expert judgment needs to be used. Lenosky et al. [44] suggest setting the weights to a fraction of the values of the data with a padding term to deal with near-zero values, so that each data point has an

unique weight. In this case, the weights are computed (with some notational change) as

$$w_m^{-1} = c_1^2 + c_2^2 \|y_m\|^2, \tag{2}$$

where c_1 and c_2 are the padding term and fractional scale of the data, respectively. The inverse weight in Eq. (2) should be understood as a permissible tolerance on each data point, with c_1 and c_2 acting as absolute and relative tolerance, respectively. In our approach, only the relative values of the weights matter, as we later scale the weights uniformly to estimate the magnitude of the model inadequacy.

The best fit parameters, $\hat{\theta}$, are those that minimize the cost function:

$$\hat{\theta} = \arg\min_{\theta} C(\theta). \tag{3}$$

There are many optimization algorithms that can be used to estimate the best fit. For machine learning potentials, stochastic gradient descent is typically used [45], while the Levenberg–Marquardt algorithm is particularly effective for training empirical potentials [26], [46], especially when the potential is sloppy (as seems to generally be the case) [47].

Having found the best fit, we quantify uncertainty in these estimated parameter values by considering sub-optimal parameter values within some tolerance. We, therefore, give a statistical interpretation to the optimization problem. The cost function is the negative log-likelihood[1] of the model parameters given the data:

$$L\left(\theta|\mathbf{y}\right) \propto \exp(-C(\theta)). \tag{4}$$

For weighted least squares, the likelihood in Eq. (4) corresponds to the assumption that the data are generated by the model with some additional, random noise:

$$y_m = f_m(\theta^*) + \epsilon_m, \tag{5}$$

where f_m is the m^{th} model prediction, θ^* are the "true" parameter values, and ϵ_m is an error term modeled as a Gaussian random variable with zero mean and variance $\sigma_m^2 = 1/w_m$ [48]. Note that the best fit parameters $\hat{\theta}$ are an estimate of θ^*.

In molecular modeling, the dominant source of error originates from the IP's functional form, i.e., it contains errors due to its limited scope and missing physics. Thus, we decompose ϵ_m in Eq. (5) as a combination of model inadequacy [7], b_m, and errors in the data (e.g., inaccuracies in the DFT values), ξ_m:

$$y_m = f_m(\theta^*) + b_m + \xi_m. \tag{6}$$

Considerable recent effort has been exerted to rigorously estimate the errors associated with DFT values [48], [49], corresponding to scale of ξ_m. However, in most molecular modeling applications, the bias b_m is the dominant source of error, and a major focus of this paper. In general, modeling the bias is an important, unsolved problem in UQ.

There are several suggestion on how to handle model bias, such as by directly improving the model or by applying statistical correction to the model prediction [38]; the latter is the focus of this paper. The statistical correction is added by inflating the likelihood [14], [48], [50], modifying Eq. (4) as

$$L(\theta|\mathbf{y}) \propto \exp(-C(\theta)/T). \tag{7}$$

Here, $T > 1$ is a hyper-parameter that we adjust to account for inadequacies of the model. Functionally, this temperature is equivalent to uniformly scaling the weights in Eq. (1). Note that the choice of T does not affect the best fit parameter values, $\hat{\theta}$, but it will affect the uncertainty associated with those values.

To estimate the statistical uncertainty in the parameters corresponding to the likelihood in Eq. (7), we use a Bayesian approach known as Markov chain Monte Carlo (MCMC). In the Bayesian framework, the parametric uncertainty is encoded in a *posterior* probability distribution of parameters given data, $P\left(\theta|\mathbf{y}\right)$. The posterior is related to the likelihood by Bayes' theorem,

$$P\left(\theta|\mathbf{y}\right) \propto L\left(\theta|\mathbf{y}\right) \times \pi(\theta), \tag{8}$$

where $L(\theta|\mathbf{y})$ is the tempered likelihood in Eq. (7) and $\pi(\theta)$ is the *prior* distribution of the parameters.

The prior, $\pi(\theta)$, must be provided by the modeler and is another important input into the UQ formalism. Nominally, it encodes the modeler's prior expectation for the values of the parameters in the model. Common choices include uniform [15], [17], [19], [21]–[24], normal [20], Jeffreys prior [16], and maximum entropy [18]. However, there is rarely an obviously "correct" prior to choose. At best, modelers may have a vague notion of a typical expected value or range for a parameter, while in other cases there may be no prior information at all. Because of this ambiguity, we recommend that calculations be done for several choices of prior distributions to ensure that any conclusions are robust to this arbitrary choice.

The remaining undefined quantity in Eq. (8) is the hyperparameter T. To choose a reasonable sampling temperature, we invoke a formal analogy between Bayesian statistics and the Boltzmann distribution in statistical mechanics. Writing the posterior as

$$P\left(\theta|\mathbf{y}\right) \propto \exp\left(-(C(\theta) - TS(\theta))/T\right) \tag{9}$$

suggests that cost is analogous to the energy while the (log) prior is analogous to the entropy, $S(\theta) = \log(\pi(\theta))$. This analogy motivates a natural way to select the temperature in Eq. (7) as an estimate of the scale of model bias. We adjust the temperature to make the fluctuations of cost in the posterior distribution comparable to the best fit cost. According to the equipartition theorem, each parameter mode in a harmonic model will contribute $T/2$, so Frederiksen et al. [14], [50] advocate using

$$T_0 = 2C_0/N, \tag{10}$$

where $C_0 = C(\hat{\theta})$ is the cost at the best fit and N is the number of parameters in the model. The value of C_0 is our best available estimate of the scale of the model's inadequacy. This choice of temperature uniformly scales the weights in

[1]The likelihood function describes the probability of obtaining the observed data for a given set of model parameters, $P\left(\mathbf{y}|\theta\right)$.

Eq. (1), inflating the error bars in the cost to be comparable in size to the minimal cost.

IPs are not harmonic models, so the specific choice given in Eq.10 is only a rough guideline [38]. Recent studies have explored anharmonic effects in sloppy IPs [13], [38]. At high sampling temperatures, the posterior is dominated by entropy and becomes sensitive to the choice of prior. The entropic contribution from the sloppy, degenerate modes can overwhelm the posterior and give biased predictions, as we demonstrate below for a specific example. Because of these anharmonic effects, practitioners should consider ensembles for many temperatures and different choices of prior to explore the sensitivity of their conclusions to these arbitrary choices.

Having defined all terms on the right-hand side of Eq. (8), the posterior $P(\theta|\mathbf{y})$ can be sampled via some MCMC-based algorithm. To efficiently sample at several temperatures, we invoke parallel-tempered MCMC (PTMCMC) methods. Tempering the likelihood as in Eq. (7) is commonly used in PTMCMC methods to improve the convergence rate of the sampling [51], [52]. These algorithms generate multiple Markov chains, each at different temperatures, and mix them with an appropriate probability to ensure convergence to the target posterior [53]. Here, we use PTMCMC methods as part of our UQ framework to empirically assess the effects of sampling temperature. In practice, we consider a chain of temperatures from $T = 1$ up to a few times larger than T_0 (defined in Eq. (10)). These temperatures explore the transition from sampling at the target accuracy (set by w_m) to a more realistic estimate of the systematic error, accounting for model inadequacy by inflating the likelihood with T_0.

After a sufficient number of iterations, the distribution of MCMC samples will converge to the posterior [54]. The multivariate potential scale reduction factor (PSRF), denoted by $\hat{R}^p$, is a common metric to assess convergence in MCMC chains. The $\hat{R}^p$ compares the variance between and within independent chains by

$$\hat{R}^p = \frac{K-1}{K} + \frac{J+1}{J}\lambda_{\max}(W^{-1}B/K), \tag{11}$$

where J and K are the numbers of chains and iterations, respectively, and $\lambda_{\max}(A)$ is the largest eigenvalue of the matrix A. The parameters, B/K and W, are the variance between and within the independent chains, ψ_j, which are calculated by

$$\begin{aligned} \frac{B}{K} &= \frac{1}{J-1}\sum_{j=1}^{J}\left(\bar{\psi}_j - \bar{\psi}\right)\left(\bar{\psi}_j - \bar{\psi}\right)^T, \\ W &= \frac{1}{J(K-1)}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(\psi_{jk} - \bar{\psi}_j\right)\left(\psi_{jk} - \bar{\psi}_j\right)^T, \end{aligned} \tag{12}$$

with ψ_{jk} denoting the k^{th} iteration of the j^{th} chain, $\bar{\psi}_j$ denoting the average of the j^{th} chain, and $\bar{\psi}$ denoting the average over all chains and iterations. The value of $\hat{R}^p$ monotonically decreases to one as $K \to \infty$ and the chains converge to the stationary distribution. In practice, a common threshold is in the range of 1.05 to 1.1 [54], [55], although higher thresholds have also been used [56].

B. The OpenKIM project

The Open Knowledgebase of Interatomic Models (OpenKIM) is a National Science Foundation (NSF)-funded cyberinfrastructure project that aims to create an organized framework for the application of IPs that yields publicly accessible and reproducible results [2]. In OpenKIM terms, a "model" refers to a standardized computer implementation of an IP with a fixed set of parameter values. OpenKIM models are publicly available in an open-source online repository at https://openkim.org/ [57]. IPs archived in OpenKIM conform to the KIM application programming interface (API) [58], which allows them to work seamlessly with multiple molecular simulation codes [59].

To facilitate the development of new IPs, the OpenKIM project has developed the KIM-based Learning-Integrated Fitting Framework (KLIFF) [60], [61]. KLIFF is a general purpose fitting framework written in Python for both physics-based and machine learning IPs. By default, KLIFF employs a force-matching algorithm to train an IP that uses the energy, forces, and stresses (if available) for a set of atomic configurations. The inclusion of other material properties in the cost function is also possible. Several built-in optimizers are provided (such as the Levenberg–Marquardt algorithm), as well as many others available through the SciPy package [62]. IPs trained with KLIFF conform to the KIM API and therefore are automatically compatible with multiple molecular simulation codes as noted above.

Finally, KLIFF is integrated with the ColabFit project [63] providing it with access to a large repository of vetted, high-quality training data for IP fitting.

III. UQ Extensions to KLIFF

The KLIFF package provides a convenient environment for fitting IPs. In this work, we introduce a new toolkit for use with KLIFF that provides a framework that facilitates transparent, reproducible UQ analysis of IPs. As MCMC is the UQ method of choice in molecular modeling, we focus on this method first with the intent to add alternative sampling methods in the future. A typical workflow is as follows: First, a modeler selects (or defines) the model and formulates the posterior sampler by instantiating `kliff.uq.MCMC`, which requires the specification of a prior and sampling temperature. Next, an MCMC simulation is performed with the `run_mcmc` function to generate ensembles of parameters drawn from the target Bayesian posterior distribution. This process is represented in Fig. 1 and further discussed below.

The first step in this workflow is to create the model. The model comprises the parameterized IP as well as atomic configurations, reference data to calibrate the IP parameters, and a simulation that calculates the corresponding material properties. KLIFF is used to construct a family of parameterized models based on an OpenKIM IP and read the configuration files that contain the atomic configurations and

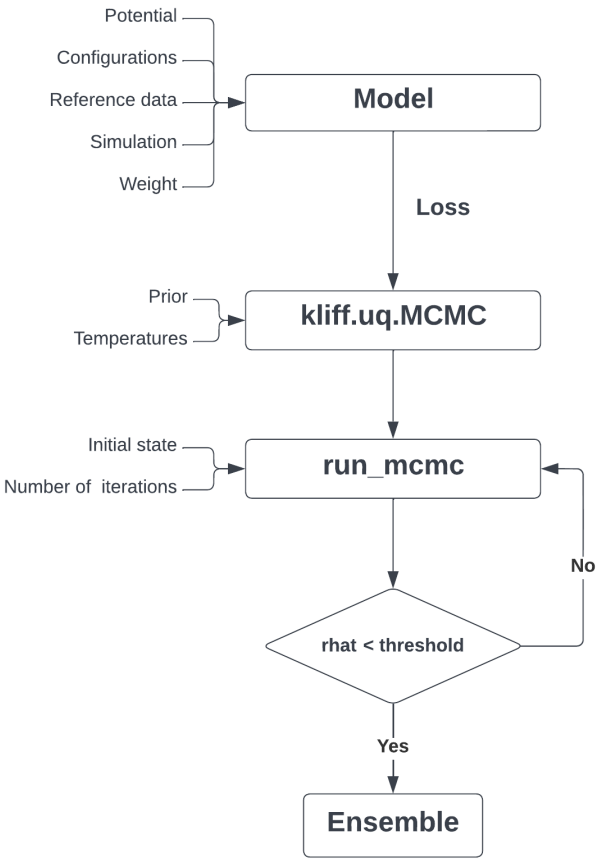

Fig. 1. UQ framework implemented in KLIFF. The UQ process starts with the construction of the model and the loss function, followed by the creation of a posterior sampler with the `kliff.uq.MCMC` class. Following that, MCMC sampling is performed with the `run_mcmc` function, terminating once the samples converge to a stationary distribution, e.g. when $\hat{R}^p$ is below an acceptable threshold. The distribution of the parameters is deduced from the samples.

reference data to calibrate the IP. By default, KLIFF uses a simulation that computes the energy, forces, and stress for each configuration, although extending KLIFF to use other simulation and the corresponding material properties is also possible (see Ref. [60]). Then, KLIFF uses this information to construct a weighted least squares cost function for the model (see Eq. (1)). An example of this process is shown in the Python script in Fig. 2.

In previous versions (version 0.3.3 or earlier), KLIFF allowed for a single weight for each type of material property in the cost function, e.g., a single weight for all energies, a single weight for all forces, and so on. To enable more flexible UQ sampling, we expand KLIFF to enable specification of custom weights for each data point. Specific support is added for the weight calculation as defined in Eq. (2) that takes arguments c_1 and c_2. The default values are $c_1 = 1.0$ and $c_2 = 0.0$, which corresponds to a uniform weight for each data point. (See `MagnitudeInverseweight` in Fig. 2 for setting c_1 and c_2 to values other than the defaults.) Alternatively, users can define their own method to compute the weights. This update is included in KLIFF version 0.4.0.

```python
import os
import numpy as np

from kliff.calculators import Calculator
from kliff.dataset import Dataset
from kliff.dataset.weight import (
    MagnitudeInverseWeight,
)
from kliff.loss import Loss
from kliff.models import KIMModel
from kliff.models.parameter_transform import (
    LogParameterTransform,
)
from kliff.utils import download_dataset

# Instantiate a transformation class to do the log
# parameter transform
param_names = ["A", "B", "sigma", "lambda", "gamma"]
params_transform = LogParameterTransform(
    param_names
)

# Instantiate the model and set the potential
model = KIMModel(
    "SW_StillingerWeber_1985_Si__MO_405512056662_006",
    params_transform,
)

# Set the tunable parameters and the initial guess
opt_params = {
    name: [["default"]] for name in param_names
}
model.set_opt_params(**opt_params)

# Get the dataset and set the weights
dataset_path = download_dataset("Si_training_set")
dataset_path /= "varying_alat"
# Instantiate the weight class
weight = MagnitudeInverseWeight(
    # Each key in weight_params contains a list
    # [c_1, c_2]
    weight_params={
        "energy_weight_params": [0.0, 0.1],
        "forces_weight_params": [1e-2, 0.1],
    }
)
# Read the configurations and reference data from
# the dataset
tset = Dataset(dataset_path, weight=weight)
configs = tset.get_configs()

# Create a calculator, which consists of simulations
# to compute material properties
calc = Calculator(model)
# Set the configurations to use by the calculator
ca = calc.create(configs)

# Instantiate the loss function
residual_data = {"normalize_by_natoms": False}
loss = Loss(calc, residual_data=residual_data)

# Train the model
loss.minimize(method="lm")
```

Fig. 2. Example Python script for constructing a model using KLIFF.

The UQ framework is implemented as module `uq` in KLIFF. The posterior sampler is constructed by creating a `kliff.uq.MCMC` instance. Internally, this action computes T_0, generates a temperature ladder, and defines the untempered log-likelihood and log-prior functions. As a default, this class inherits from the sampler in the `ptemcee` Python package to perform PTMCMC. The algorithm simulates multiple chains (several at each sampling temperature) in parallel and mixes chains from different sampling temperatures to allow the MCMC walkers to explore a wider range of parameters. The number of chains or walkers can be specified through an optional argument `nwalkers`; the default value is twice the number of parameters in the model. This process is illustrated in the listing in Fig. 3.

```
from kliff.uq import MCMC, get_T0
from multiprocessing import Pool

np.random.seed(2022)

# Get the dimensionality of the problem
# Number of parameters
ndim = calc.get_num_opt_params()
nwalkers = 2 * ndim  # Number of parallel walkers

# Generate a temperature ladder
T0 = get_T0(loss)
Tladder = np.sort(
    np.append(np.logspace(0, 7, 15), T0)
)
ntemps = len(Tladder)  # Number of temperatures

# Instantiate a sampler
sampler = MCMC(
    loss,
    nwalkers=nwalkers,
    logprior_args=(np.tile([-8, 8], (ndim, 1)),),
    Tladder=Tladder,
    # Other keyword arguments for ptemcess.Sampler
    random=np.random.RandomState(2022),
)
# Declare multiprocessing pool for parallel
# computing
sampler.pool = Pool(processes=nwalkers)
```

Fig. 3. Example Python script for constructing the sampler with the `kliff.uq.MCMC` class. See Fig. 2 for an example of how to define the calculator and loss function.

The arguments to instantiate `kliff.uq.MCMC` are (1) a `kliff.loss.Loss` instance, which defines the *untempered* ($T = 1$) likelihood function in Eq. (4), (2) the prior, and (3) the sampling temperatures. Since uniform priors are a common choice, the constructor implements this by default. In this case, the boundaries of the prior support need to be specified by the user via the `logprior_args` argument. However, the user can also pass a custom prior as the `logprior_fn` argument.

There are two options to specify the temperature ladder. First, the user may specify the number of temperatures (`ntemps`) and the ratio between the maximum temperature and T_0 in Eq. (10) (`Tmax_ratio`). In this case, an internal function generates a logarithmically spaced temperature between $T = 1.0$ and $T = T_{\text{max\_ratio}} \times T_0$, inclusive. Alternatively, the user may specify the complete list of temperature values (`Tladder`). When `Tladder` is specified, then this list overrides the values passed for `ntemps` and `Tmax_ratio`.

The UQ implementation in KLIFF supports parallelization over the configurations for the cost function evaluation and over the walkers for the MCMC sampling. However, the current implementation only supports OpenMP-style parallelization for the cost function evaluation and both OpenMP and MPI for the MCMC sampling, with a future work to allow MPI in the earlier. In this paper we only show an example of parallelization in the Bayesian sampling, which is done by declaring a `multiprocessing` pool after instantiating `kliff.uq.MCMC`. The optimal number of parallel processes in this case is the product of the numbers of the sampling temperatures and the walkers.

After constructing the posterior sampler, MCMC sampling is run by calling the `run_mcmc` method of the `kliff.uq.MCMC` class (see Fig. 4). This function requires the initial position of each walker as an $L \times J \times N$ array, where L, J, and N are the number of temperatures, walkers, and parameters, respectively. The user also needs to specify the number of iterations to run the MCMC simulation.

```
# Initial starting points for each walker
p0 = np.random.uniform(
    low=-6.0,
    high=6.0,
    size=(ntemps, nwalkers, ndim),
)

# Run MCMC
sampler.run_mcmc(p0, 150000)
sampler.pool.close()
```

Fig. 4. Example Python script for using the `run_mcmc` function to perform sampling.

The convergence of the parameter chains is assessed by calculating $\hat{R}^p$. In our implementation, this is realized by the function `kliff.uq.rhat`, demonstrated by the listing in Fig. 5. This function takes an array containing the MCMC samples for one sampling temperature. The $\hat{R}^p$ is calculated for each sampling temperature separately. Note that in practice, some sampling temperatures may converge much sooner than others. If the resulting values are larger than some threshold (typically 1.1), then the MCMC algorithm should continue to iterate. If $\hat{R}^p$ is less than the target threshold, the samples are assumed to have converged to the posterior and the calculation is terminated.

This UQ framework is integrated in KLIFF and we provide some examples in an online repository [64]. In the next section, we describe the use and interpretation of a UQ calculation for a Stillinger–Weber potential.

IV. Results

Having described the basic interface to this UQ toolkit, we now consider the concrete example given in the combined listings in Figs. 2–5. In this section we discuss the specific

```python
from kliff.uq import rhat

# Retrieve the samples
# Set the burn-in time and thinning factor
burnin, thin = 10000, 200
samples = sampler.chain[:, :, burnin::thin]

# Assess convergence by computing rhat
rhat_array = np.empty(ntemps)
for tidx in range(ntemps):
    rhat_array[tidx] = rhat(samples[tidx])
```

Fig. 5. Example Python script for using `kliff.uq.rhat` function to compute $\hat{R}^p$ as a convergence assessment tool.

model and MCMC setup in these scripts. We then present and discuss the sampling results and some of the subtleties associated with their interpretation.

This example is based on the Stillinger–Weber potential in OpenKIM [65], [66]. This is a cluster potential originally introduced to model silicon [67]. For a system with n atoms, the potential energy of atom i, $\mathcal{V}_i$, is given by

$$\mathcal{V}_i = \sum_{j>i}^{n} \phi_2(r_{ij}) + \sum_{j\neq i}^{n} \sum_{\substack{k>j \\ k\neq i}}^{n} \phi_3(r_{ij}, r_{ik}, \beta_{jik}), \tag{13}$$

where ϕ_m denotes the m-body interactions. The Stillinger–Weber potential includes both two-body and three-body interactions. The two-body term only depends on the distance between atom i and j, denoted by r_{ij},

$$\phi_2(r_{ij}) = A\left[B\left(\frac{\sigma}{r_{ij}}\right)^p - \left(\frac{\sigma}{r_{ij}}\right)^q\right] \times \exp\left(\frac{\sigma}{r_{ij} - r^{\text{cut}}}\right). \tag{14}$$

The three-body term additionally depends on β_{ijk}, which is the bond angle between the $i-j$ and $i-k$ bonds,

$$\phi_3\left(r_{ij}, r_{ik}, \beta_{jik}\right) = \lambda\left(\cos\beta_{jik} - \cos\beta^0\right)^2 \times \exp\left(\frac{\gamma}{r_{ij} - r^{\text{cut}}} + \frac{\gamma}{r_{ik} - r^{\text{cut}}}\right). \tag{15}$$

This IP includes nine parameters: A, B, σ, p, q, r^{cut}, λ, β^0 and γ. The total energy of the system $\mathcal{V}$ is

$$\mathcal{V} = \sum_{i=1}^{n} \mathcal{V}_i. \tag{16}$$

The atomic forces are calculated by taking the negative gradient of Eq. (13) with respect to the atomic positions.

In this example, we choose the tunable parameters to be $A, B, \sigma, \lambda,$ and γ. The other parameters are fixed to the default values given in OpenKIM [66]. Physically, the tunable parameters A and λ set energy scales in the potential, and σ and γ set length scales. To be physically relevant, these parameters are constrained to be positive. Parameter B controls the relative scale of the repulsive part of the interaction and, thus, is also constrained to be positive. To enforce this constraint, we use a log-transform, i.e., the tunable parameters are $\theta = (\log(A), \log(B), \log(\sigma), \log(\lambda), \log(\gamma))$.

The training set consists of the energies and forces of silicon in the diamond cubic crystal structure. We use 400 atomic configurations, including stretched and compressed cells with random perturbations. The reference energy and force data were generated using the environment-dependent interatomic potential (EDIP) [68]–[70].[2] Next, we compute the weights using Eq. (2), with $c_1 = 10^{-2}$ eV/Å for the forces and zero for the energies, and $c_2 = 10^{-1}$ for both.

The training produces the following best fit parameters for the SW potential:

$$A = 15.27922231 \text{ eV} \qquad \lambda = 45.47927476 \text{ eV}$$
$$B = 0.6032372 \qquad \gamma = 2.51306949 \text{ Å}$$
$$\sigma = 2.09420085 \text{ Å}.$$

The natural temperature for this model is $T_0 = 1.324$. Notice that the reference data were generated using another IP. Thus, while there is some model inadequacy, it is relatively small compared to some other real-world examples, which explains the low T_0 value; in practice T_0 is typically orders of magnitude larger [13]. In our analysis, we extend the temperature ladder to a much higher temperature to mimic typical effects users may encounter in real-world applications.

We use a uniform prior where $\pi(\theta)$ is constant if $-8 < \theta < 8$ and zero otherwise. The support of this prior is chosen to be sufficiently wide to ensure that sampling is not artificially restricted to regions near the best fit.

Next, PTMCMC sampling is performed for 150,000 iterations. From each walker, we discard the first 10,000 steps as the burn-in time. We thin the remaining chains by keeping every 200-th step to ensure uncorrelated samples. From the remaining samples, the maximum value of $\hat{R}^p$ is 1.046.

The posterior distribution from which samples are drawn is a joint distribution for the five parameters. Because we cannot directly visualize such a high dimensional space, it is common to instead plot marginal distributions. The marginal distribution of a high-dimensional, joint probability distribution is the projection of the probability onto a lower dimensional subspace.[3] We project the five-dimensional posterior probability distribution onto each of the one-dimensional parameter axes in Fig. 6. The marginal distributions at different sampling temperatures are shown in different colors and superimposed to enable direct comparison. Each column corresponds to a different parameter in the potential. In the second row, a log scale is used for the vertical axis to bring out the details of the sparser distributions that result at higher sampling temperatures.

Examining Fig. 6, first note the general trend that distributions becomes wider as the temperature increases. This is expected, since higher temperatures correspond to smaller

[2]Since our objective is to explore UQ rather than develop an accurate model for silicon, we take the EDIP potential to be the "exact" ground truth. This greatly reduces the computational cost of generating the training set relative to DFT.

[3]The marginal distribution of a parameter is calculated by summing the conditional distributions of that parameter over all possible values of the other parameters.

weights, i.e., larger effective error bars, in Eq. (1). However, the effect is parameter-dependent. For example, consider the distributions of $\log(\lambda)$ and $\log(\gamma)$ at $T = 10^2$. At this temperature, the distributions are relatively localized around their best fit values. However, at the next highest temperature, $T = 10^3$, the distributions extend to the boundary of their respective priors. This phenomenon, in which the marginal posterior distribution of some parameters abruptly transitions away from being localized at a specific sampling temperature, is called *parameter evaporation*. Inspecting Fig. 6 reveals that $\log(\lambda)$ and $\log(\gamma)$ evaporate around $T = 10^3$ while parameters A and B evaporate around $T = 10^4$. At a sufficiently high sampling temperature, all parameters would evaporate, regardless of the prior, as show in Ref. [13] for SW potential for a molybdenum disulfide system.

Parameter evaporation has been observed in Ref. [13] for IPs and has important implications for UQ analysis. To see this, first notice that while the distribution for A and B remain localized at $T = 10^3$, they have shifted relative to their distributions at $T = 10^2$, i.e., they are localized around different values. We can explain this shift in terms of the evaporation of the other parameters. At $T = 10^3$, the range of values sampled for the parameters λ and γ are very different from those at $T = 10^2$. Consequently, the sampled values for the parameters A and B shift to minimize the cost with this more diffuse distribution of λ and γ. That is to say, the values for parameters A and B are biased as a consequence of having the sub-optimal values of λ and γ. While this shift is not inherently problematic, notice that the distributions for A and B at $T = 10^3$ have very little overlap with their counterparts at $T = 10^2$. Because the lower temperature distribution is dominated by samples near the best fit parameters, we infer that the higher temperature distribution has very few samples near the best fit. This is potentially problematic since it implies that parameter values that best fit the data are not represented in the sample.

Figure 7 shows the distribution of the untempered costs at each sampling temperature. As expected, the average value of the cost increases at higher temperatures. However, this increase is not the result of stretching the distribution, as is the case for linear regression model. Rather, the entire distribution shifts to the right at each rung in the temperature ladder. As the temperature increases and parameters evaporate, the posterior is dominated by regions of the parameter space that are poor fits to the data. These are regions of parameter space in which the prior places considerable weight, that is, they are high-entropy regions. This is a nonlinear effect due to the subtle interplay between the sampling temperature, the prior, and the degenerate modes of sloppy models. The resulting posterior distribution can depend very strongly on details of the problem, such as the choice of prior, the error bars for the data, and the sampling temperature.

In general, we recommend that practitioners check their results for robustness for several priors over a range of temperatures. In this work, we have used uniform priors in log-transformed parameters as a pedagogical example. This is a reasonable choice in general; however, we could have also used uniform priors in the original, untransformed parameterization as well as Gaussian priors in both log-transformed and untransformed parameters. We caution against using Jeffreys prior, as its interaction with the degenerate modes can lead to strong biases [71]. In this example, we have sampled up to a relatively high temperature relative to T_0 as defined in Eq. (10). This was also a pedagogical choice to illustrate important effects and mimic more realistic examples (for example, previous work has seen values of $T_0 > 10^6$ [13]). In practice, we suggest including sampling temperatures up to a few times larger than T_0. The highest sampling temperatures are not for the purposes of UQ, but including them in the sampling algorithm improves convergence rates. For UQ analysis, we suggest considering ensembles generated by temperatures from 50% below to 50% above T_0.

IP developers should also use uncertainty quantification tools throughout the model development cycle. For example, they could extend the training data to better constrain the sloppy parameters. Alternatively, they could use model reduction methods [72] to remove the degenerate modes from the model. For the latter, care must be taken to not remove unidentifiable parameters that would be important for downstream applications.

V. Conclusion

In this work, we describe a UQ toolkit that extends the KLIFF package as part of the OpenKIM environment. This toolkit provides a framework that facilitates transparent, reproducible UQ analysis for both the development and application of IPs in molecular modeling. We focus on a Bayesian, MCMC approach for quantifying parametric uncertainty and model inadequacy. We use a parallel-tempered MCMC method, which improves convergence and allows us to mimic the effect of model inadequacy in our uncertainty estimates.

In our implementation we focused on ease of use in order to lower the barrier to entry for molecular modeling practitioners. However, we caution users not to treat these methods as off-the-shelf black boxes. UQ is an emerging field with many open questions, especially surrounding model inadequacy which is often the dominant problem in atomistic simulations. We encourage IP practitioners and developers alike to familiarize themselves with statistical subtleties related to sloppiness and parameter unidentifiability [13], [47] and check their conclusions for robustness over a range of sampling temperatures for multiple priors.

In future work, we plan to integrate other UQ methods within KLIFF, such as the frequentist profile likelihood [73]. To address the problems associated with UQ of sloppy IPs [13], we plan to integrate a model reduction scheme motivated by information geometry [72].

VI. Acknowledgment

This work is supported by the National Science Foundation under awards DMR-1834332 and DMR-1834251. We would

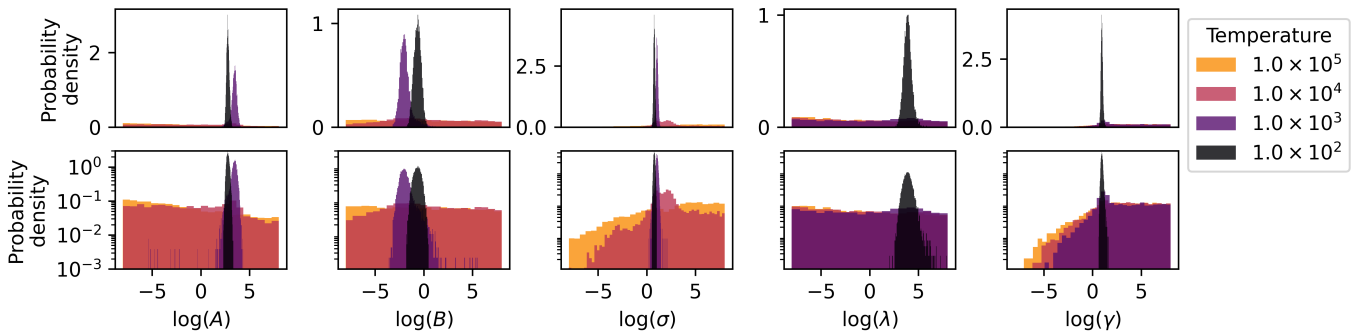

Fig. 6. Marginal distributions of the MCMC samples (the projection of the joint distribution onto a single parameter axis) of the SW potential at several sampling temperatures, normalized by the number of samples. Each column shows the distribution of a each parameter, with the sampling temperatures shown by the different colors. On the second row, the distributions are presented in logarithmic scale on the vertical axis to bring out the details at higher sampling temperatures.

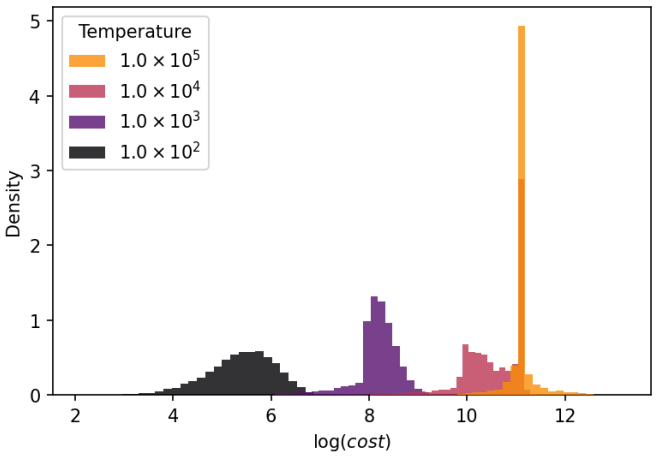

Fig. 7. Distribution of cost at each sampling temperature.

like to acknowledge the computational facilities provided by the Brigham Young University Office of Research Computing.

References

[1] D. W. Brenner, "The art and science of an analytic potential," *physica status solidi (b)*, vol. 217, no. 1, pp. 23–40, 2000.

[2] E. B. Tadmor, R. S. Elliott, J. P. Sethna, R. E. Miller, and C. A. Becker, "The potential of atomistic simulations and the Knowledgebase of Interatomic Models," *JOM*, vol. 63, no. 7, pp. 17–17, Jul 2011.

[3] E. v. d. Giessen, P. A. Schultz, N. Bertin, V. V. Bulatov, W. Cai, G. Csányi, S. M. Foiles, M. G. D. Geers, C. González, M. Hütter, W. K. Kim, D. M. Kochmann, J. LLorca, A. E. Mattsson, J. Rottler, A. Shluger, R. B. Sills, I. Steinbach, A. Strachan, and E. B. Tadmor, "Roadmap on multiscale materials modeling," *Modelling and Simulation in Materials Science and Engineering*, vol. 28, no. 4, p. 043001, Mar 2020.

[4] R. LeSar, *Introduction to computational materials science*. Cambridge, England: Cambridge University Press, Apr. 2013.

[5] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, E. D. Hermes, P. C. Jennings, P. B. Jensen, J. Kermode, J. R. Kitchin, E. L. Kolsbjerg, J. Kubal, K. Kaasbjerg, S. Lysgaard, J. B. Maronsson, T. Maxson, T. Olsen, L. Pastewka, A. Peterson, C. Rostgaard, J. Schiøtz, O. Schütt, M. Strange, K. S. Thygesen, T. Vegge, L. Vilhelmsen, M. Walter, Z. Zeng, and K. W. Jacobsen, "The atomic simulation environment – a Python library for working with atoms," *Journal of Physics: Condensed Matter*, vol. 29, no. 27, p. 273002, Jun 2017.

[6] A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, and S. J. Plimpton, "Lammps - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales," *Computer Physics Communications*, vol. 271, p. 108171, Feb 2022.

[7] M. C. Kennedy and A. O'Hagan, "Bayesian calibration of computer models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 3, p. 425464, 2001.

[8] F. Ercolessi and J. B. Adams, "Interatomic potentials from first-principles calculations: The force-matching method," *EPL (Europhysics Letters)*, vol. 26, no. 8, p. 583, Jun 1994.

[9] D. S. Karls, "Transferability of empirical potentials and the Knowledgebase of Interatomic Models (KIM)," Ph.D., University of Minnesota, United States – Minnesota, Apr 2016. [Online]. Available: http://www.proquest.com/docview/1822512470/abstract/22AAC5589C4D49E6PQ/1

[10] R. A. Messerly, T. A. Knotts, and W. V. Wilding, "Uncertainty quantification and propagation of errors of the Lennard-Jones 12-6 parameters for n-alkanes," *The Journal of Chemical Physics*, vol. 146, no. 19, p. 194110, May 2017.

[11] M. A. Tschopp, B. Chris Rinderspacher, S. Nouranian, M. I. Baskes, S. R. Gwaltney, and M. F. Horstemeyer, "Quantifying parameter sensitivity and uncertainty for interatomic potential design: Application to saturated hydrocarbons," *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg*, vol. 4, no. 1, Mar 2018. [Online]. Available: https://asmedigitalcollection.asme.org/risk/article/4/1/011004/370001/Quantifying-Parameter-Sensitivity-and-Uncertainty

[12] A. Mishra, S. Hong, P. Rajak, C. Sheng, K.-i. Nomura, R. K. Kalia, A. Nakano, and P. Vashishta, "Multiobjective genetic training and uncertainty quantification of reactive force fields," *npj Computational Materials*, vol. 4, no. 11, pp. 1–7, Aug 2018.

[13] Y. Kurniawan, C. L. Petrie, K. J. Williams, M. K. Transtrum, E. B. Tadmor, R. S. Elliott, D. S. Karls, and M. Wen, "Bayesian, frequentist, and information geometry approaches to parametric uncertainty quantification of classical empirical interatomic potentials," *arXiv preprint arXiv:2112.10851*, 2021.

[14] S. L. Frederiksen, K. W. Jacobsen, K. S. Brown, and J. P. Sethna, "Bayesian ensemble approach to error estimation of interatomic potentials," *Physical Review Letters*, vol. 93, no. 16, p. 165501, Oct 2004.

[15] P. Angelikopoulos, C. Papadimitriou, and P. Koumoutsakos, "Bayesian uncertainty quantification and propagation in molecular dynamics simulations: A high performance computing framework," *The Journal of Chemical Physics*, vol. 137, no. 14, p. 144103, Oct 2012.

[16] F. Rizzi, H. N. Najm, B. J. Debusschere, K. Sargsyan, M. Salloum, H. Adalsteinsson, and O. M. Knio, "Uncertainty quantification in MD simulations. part II: Bayesian inference of force-field parameters," *Multiscale Modeling & Simulation*, vol. 10, no. 4, pp. 1460–1492, Jan 2012.

[17] F. Rizzi, R. E. Jones, B. J. Debusschere, and O. M. Knio, "Uncertainty quantification in MD simulations of concentration driven ionic flow

through a silica nanopore. II. uncertain potential parameters," *The Journal of Chemical Physics*, vol. 138, no. 19, p. 194105, May 2013.
[18] M. Cools-Ceuppens and T. Verstraelen, "Uncertainty prediction in molecular simulations using ab initio derived force fields." Ph.D. dissertation, Ghent University, 2017. [Online]. Available: https://lib.ugent.be/fulltxt/RUG01/002/366/973/RUG01-002366973_2017_0001_AC.pdf
[19] R. A. Messerly, M. R. Shirts, and A. F. Kazakov, "Uncertainty quantification confirms unreliable extrapolation toward high pressures for united-atom Mie λ-6 force field," *The Journal of Chemical Physics*, vol. 149, no. 11, p. 114109, Sep 2018.
[20] L. De Simon, M. Iglesias, B. Jones, and C. Wood, "Quantifying uncertainty in thermophysical properties of walls by means of Bayesian inversion," *Energy and Buildings*, vol. 177, pp. 220–245, Oct 2018.
[21] M. Vohra, A. Y. Nobakht, S. Shin, and S. Mahadevan, "Uncertainty quantification in non-equilibrium molecular dynamics simulations of thermal transport," *International Journal of Heat and Mass Transfer*, vol. 127, pp. 297–307, Dec 2018.
[22] M. Vohra and S. Mahadevan, "Discovering the active subspace for efficient UQ of molecular dynamics simulations of phonon transport in silicon," *International Journal of Heat and Mass Transfer*, vol. 132, pp. 577–586, Apr 2019.
[23] G. Dhaliwal, P. B. Nair, and C. V. Singh, "Uncertainty analysis and estimation of robust AIREBO parameters for graphene," *Carbon*, vol. 142, pp. 300–310, Feb 2019.
[24] ——, "Uncertainty and sensitivity analysis of mechanical and thermal properties computed through Embedded Atom Method potential," *Computational Materials Science*, vol. 166, pp. 30–41, Aug 2019.
[25] S. Longbottom and P. Brommer, "Uncertainty quantification for classical effective potentials: an extension to potfit," *Modelling and Simulation in Materials Science and Engineering*, vol. 27, no. 4, p. 044001, Apr 2019.
[26] M. Wen, J. Li, P. Brommer, R. S. Elliott, J. P. Sethna, and E. B. Tadmor, "A KIM-compliant potfit for fitting sloppy interatomic potentials: application to the EDIP model for silicon," *Modelling and Simulation in Materials Science and Engineering*, vol. 25, no. 1, p. 014001, Nov 2016.
[27] M. Wen, S. N. Shirodkar, P. Plecháč, E. Kaxiras, R. S. Elliott, and E. B. Tadmor, "A force-matching stillinger-weber potential for MoS_2: Parameterization and fisher information theory based sensitivity analysis," *Journal of Applied Physics*, vol. 122, no. 24, p. 244301, Dec 2017.
[28] J. E. Jeong, Q. Zhuang, M. K. Transtrum, E. Zhou, and P. Qiu, "Experimental design and model reduction in systems biology," *Quantitative Biology*, vol. 6, no. 4, pp. 287–306, Dec 2018.
[29] A. White, M. Tolman, H. D. Thames, H. R. Withers, K. A. Mason, and M. K. Transtrum, "The limitations of model-based experimental design and parameter estimation in sloppy systems," *PLOS Computational Biology*, vol. 12, no. 12, p. e1005227, Dec 2016.
[30] M. K. Transtrum and P. Qiu, "Bridging mechanistic and phenomenological models of complex biological systems," *PLOS Computational Biology*, vol. 12, no. 5, p. e1004915, May 2016.
[31] B. K. Mannakee, A. P. Ragsdale, M. K. Transtrum, and R. N. Gutenkunst, *Sloppiness and the Geometry of Parameter Space*, ser. Studies in Mechanobiology, Tissue Engineering and Biomaterials. Springer International Publishing, 2016, vol. 17, pp. 271–299. [Online]. Available: http://link.springer.com/10.1007/978-3-319-21296-8_11
[32] M. K. Transtrum, B. B. Machta, K. S. Brown, B. C. Daniels, C. R. Myers, and J. P. Sethna, "Perspective: Sloppiness and emergent theories in physics, biology, and beyond," *The Journal of Chemical Physics*, vol. 143, no. 1, p. 010901, Jul 2015.
[33] M. K. Transtrum and P. Qiu, "Optimal experiment selection for parameter estimation in biological differential equation models," *BMC Bioinformatics*, vol. 13, no. 1, p. 181, Jul 2012.
[34] M. K. Transtrum, A. T. Sarić, and A. M. Stanković, "Information geometry approach to verification of dynamic models in power systems," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 440–450, 2017.
[35] R. Gutenkunst, "Sloppiness, modeling, and evolution in biochemical networks," Ph.D. dissertation, Cornell University, Aug 2007, accepted: 2007-08-29T17:25:45Z. [Online]. Available: https://ecommons.cornell.edu/handle/1813/8206
[36] B. B. Machta, R. Chachra, M. K. Transtrum, and J. P. Sethna, "Parameter space compression underlies emergent theories and predictive models," *Science*, vol. 342, no. 6158, pp. 604–607, Nov 2013.
[37] J. J. Waterfall, F. P. Casey, R. N. Gutenkunst, K. S. Brown, C. R. Myers, P. W. Brouwer, V. Elser, and J. P. Sethna, "Sloppy-model universality class and the Vandermonde matrix," *Phys. Rev. Lett.*, vol. 97, p. 150601, Oct 2006. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.97.150601
[38] P. Pernot, "The parameter uncertainty inflation fallacy," *The Journal of Chemical Physics*, vol. 147, no. 10, p. 104102, Sep 2017.
[39] D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman, "emcee: The mcmc hammer," *Publications of the Astronomical Society of the Pacific*, vol. 125, no. 925, pp. 306–312, Mar 2013, arXiv: 1202.3665.
[40] J. Feinberg and H. P. Langtangen, "Chaospy: An open source tool for designing methods of uncertainty quantification," *Journal of Computational Science*, vol. 11, pp. 46–57, Nov 2015.
[41] R. A. Richardson, D. W. Wright, W. Edeling, V. Jancauskas, J. Lakhlili, and P. V. Coveney, "Easyvvuq: A library for verification, validation and uncertainty quantification in high performance computing," *Journal of Open Research Software*, vol. 8, no. 11, p. 11, Apr 2020.
[42] P. Brommer and F. Gähler, "Potfit: effective potentials from *ab initio* data," *Modelling and Simulation in Materials Science and Engineering*, vol. 15, no. 3, pp. 295–304, mar 2007. [Online]. Available: https://doi.org/10.1088/0965-0393/15/3/008
[43] M. R. Fellinger, H. Park, and J. W. Wilkins, "Force-matched embedded-atom method potential for niobium," *Physical Review B*, vol. 81, no. 14, p. 144119, Apr 2010.
[44] T. J. Lenosky, J. D. Kress, I. Kwon, A. F. Voter, B. Edwards, D. F. Richards, S. Yang, and J. B. Adams, "Highly optimized tight-binding model of silicon," *Physical Review B*, vol. 55, no. 3, pp. 1528–1544, Jan 1997.
[45] J. Behler, "Perspective: Machine learning potentials for atomistic simulations," *The Journal of Chemical Physics*, vol. 145, no. 17, p. 170901, Nov 2016.
[46] M. K. Transtrum and J. P. Sethna, "Improvements to the Levenberg-Marquardt algorithm for nonlinear least-squares minimization," *arXiv:1201.5885 [physics]*, Jan 2012, arXiv: 1201.5885. [Online]. Available: http://arxiv.org/abs/1201.5885
[47] M. K. Transtrum, B. B. Machta, and J. P. Sethna, "Why are nonlinear fits so challenging?" *Physical Review Letters*, vol. 104, no. 6, p. 060201, Feb 2010, arXiv: 0909.3884.
[48] R. Christensen, T. Bligaard, and K. W. Jacobsen, *3 - Bayesian error estimation in density functional theory*, ser. Elsevier Series in Mechanics of Advanced Materials. Woodhead Publishing, Jan 2020, pp. 77–91. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780081029411000031
[49] J. J. Mortensen, K. Kaasbjerg, S. L. Frederiksen, J. K. Nørskov, J. P. Sethna, and K. W. Jacobsen, "Bayesian error estimation in density-functional theory," *Physical Review Letters*, vol. 95, no. 21, p. 216401, Nov 2005.
[50] V. Petzold, T. Bligaard, and K. W. Jacobsen, "Construction of new electronic density functionals with error estimation through fitting," *Topics in Catalysis*, vol. 55, no. 5, p. 402417, Jun 2012.
[51] D. J. Earl and M. W. Deem, "Parallel tempering: Theory, applications, and new perspectives," *Physical Chemistry Chemical Physics (Incorporating Faraday Transactions)*, vol. 7, p. 3910, 2005.
[52] J. W. Miller and D. B. Dunson, "Robust bayesian inference via coarsening," *Journal of the American Statistical Association*, vol. 114, no. 527, p. 11131125, Jul 2019.
[53] W. Vousden, W. M. Farr, and I. Mandel, "Dynamic temperature selection for parallel-tempering in markov chain monte carlo simulations," *Monthly Notices of the Royal Astronomical Society*, vol. 455, no. 2, pp. 1919–1937, Jan 2016, arXiv: 1501.05823.
[54] A. Gelman and D. B. Rubin, "Inference from iterative simulation using multiple sequences," *Statistical Science*, vol. 7, no. 4, pp. 457–472, Nov 1992, zbl: 06853057.
[55] S. P. Brooks and A. Gelman, "General methods for monitoring convergence of iterative simulations," *Journal of Computational and Graphical Statistics*, vol. 7, no. 4, pp. 434–455, Dec 1998.
[56] D. Vats and C. Knudson, "Revisiting the Gelman–Rubin diagnostic," *Statistical Science*, vol. 36, no. 4, pp. 518–529, 2021.
[57] "Types of kim content." [Online]. Available: https://openkim.org/doc/repository/kim-content/
[58] R. S. Elliott and E. B. Tadmor, "Knowledgebase of Interatomic Models (KIM) application programming interface (API)," https://openkim.org/kim-api, 2011.
[59] "Simulation codes compatible with the KIM API." [Online]. Available: https://openkim.org/projects-using-kim/

[60] M. Wen, Y. Afshar, R. S. Elliott, and E. B. Tadmor, "KLIFF: A framework to develop physics-based and machine learning interatomic potentials," *Computer Physics Communications*, vol. 272, p. 108218, Mar 2022.
[61] "KLIFF git repository." [Online]. Available: https://github.com/openkim/kliff
[62] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.
[63] "Colabfit: Collaborative development of data-driven interatomic potentials for predictive molecular simulations." [Online]. Available: https://colabfit.org/
[64] Y. Kurniawan, "KLIFF_uq." [Online]. Available: https://gitlab.com/yonatank93/kliff_uq
[65] M. Wen, Y. Afshar, F. H. Stillinger, and T. A. Weber, "Stillinger-Weber (SW) Model Driver v005," OpenKIM, https://doi.org/10.25950/dd263fe3, 2021.
[66] A. K. Singh, F. H. Stillinger, and T. A. Weber, "Stillinger-Weber potential for Si due to Stillinger and Weber (1985) v006," OpenKIM, https://doi.org/10.25950/dd263fe3, 2021.
[67] F. H. Stillinger and T. A. Weber, "Computer simulation of local order in condensed phases of silicon," *Physical Review B*, vol. 31, pp. 5262–5271, Apr 1985.
[68] D. S. Karls, J. F. Justo, M. Z. Bazant, E. Kaxiras, V. V. Bulatov, and S. Yip, "Environment-Dependent Interatomic Potential (EDIP) model driver v002," OpenKIM, https://doi.org/10.25950/545ca247, 2018.
[69] D. S. Karls, "EDIP model for Si developed by Justo et al. (1998) v002," OpenKIM, https://doi.org/10.25950/545ca247, 2018.
[70] J. a. F. Justo, M. Z. Bazant, E. Kaxiras, V. V. Bulatov, and S. Yip, "Interatomic potential for silicon defects and disordered phases," *Physical Review B*, vol. 58, pp. 2539–2550, Aug 1998.
[71] K. N. Quinn, M. C. Abbott, M. K. Transtrum, B. B. Machta, and J. P. Sethna, "Information geometry for multiparameter models: New perspectives on the origin of simplicity," *arXiv preprint arXiv:2111.07176*, 2021.
[72] M. K. Transtrum and P. Qiu, "Model reduction by manifold boundaries," *Physical Review Letters*, vol. 113, no. 9, p. 098701, Aug 2014.
[73] S. R. Cole, H. Chu, and S. Greenland, "Maximum likelihood, profile likelihood, and penalized likelihood: A primer," *American Journal of Epidemiology*, vol. 179, no. 2, pp. 252–260, Jan 2014.