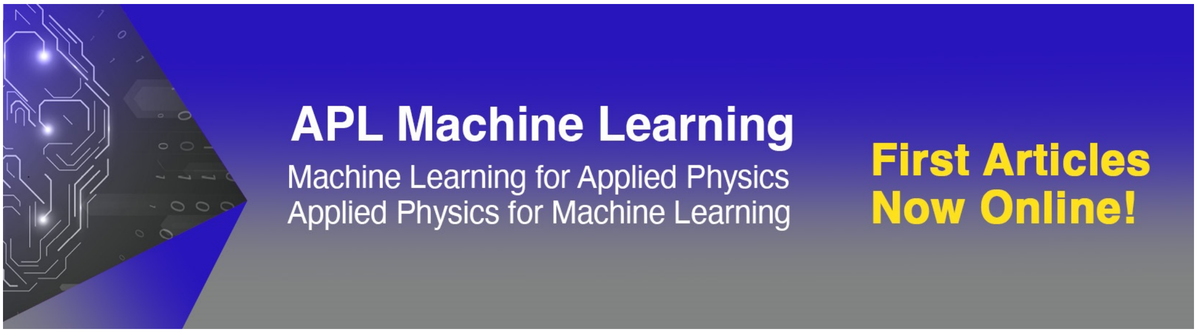
The Journal of Chemical Physics

TBMaLT, a flexible toolkit for combining tight-binding and machine learning

Cite as: J. Chem. Phys. **158**, 034801 (2023); https://doi.org/10.1063/5.0132892
Submitted: 31 October 2022 • Accepted: 02 January 2023 • Accepted Manuscript Online: 02 January 2023 • Published Online: 18 January 2023

A. McSloy, G. Fan, W. Sun, et al.


APL Machine Learning
Machine Learning for Applied Physics
Applied Physics for Machine Learning
First Articles Now Online!

AIP Publishing

The Journal of Chemical Physics ARTICLE scitation.org/journal/jcp

TBMaLT, a flexible toolkit for combining tight-binding and machine learning

Cite as: J. Chem. Phys. 158, 034801 (2023); doi: 10.1063/5.0132892
Submitted: 31 October 2022 • Accepted: 2 January 2023 •
Published Online: 18 January 2023

A. McSloy,[1,a)] G. Fan,[2] W. Sun,[2] C. Hölzer,[3] M. Friede,[3] S. Ehlert,[3,4] N.-E. Schütte,[2] S. Grimme,[3] T. Frauenheim,[2,5] and B. Aradi[2,b)]

AFFILIATIONS

[1] Warwick Centre for Predictive Modelling, School of Engineering, University of Warwick, Coventry CV4 7AL, United Kingdom
[2] Bremen Center of Computational Materials Science, University of Bremen, 28359 Bremen, Germany
[3] Mulliken Center for Theoretical Chemistry, University of Bonn, 53115 Bonn, Germany
[4] Microsoft Research AI4Science, 1118 CZ Schiphol, Netherlands
[5] Computational Science Research Center (CSRC) Beijing and Computational Science Applied Research (CSAR) Institute Shenzhen, Shenzhen, China

Note: This paper is part of the JCP Special Topic on Modern Semiempirical Electronic Structure Methods.
[a)] **Electronic mail:** adam.mcsloy@warwick.ac.uk
[b)] **Author to whom correspondence should be addressed:** aradi@uni-bremen.de

ABSTRACT

Tight-binding approaches, especially the Density Functional Tight-Binding (DFTB) and the extended tight-binding schemes, allow for efficient quantum mechanical simulations of large systems and long-time scales. They are derived from *ab initio* density functional theory using pragmatic approximations and some empirical terms, ensuring a fine balance between speed and accuracy. Their accuracy can be improved by tuning the empirical parameters using machine learning techniques, especially when information about the local environment of the atoms is incorporated. As the significant quantum mechanical contributions are still provided by the tight-binding models, and only short-ranged corrections are fitted, the learning procedure is typically shorter and more transferable as it were with predicting the quantum mechanical properties directly with machine learning without an underlying physically motivated model. As a further advantage, derived quantum mechanical quantities can be calculated based on the tight-binding model without the need for additional learning. We have developed the open-source framework—Tight-Binding Machine Learning Toolkit—which allows the easy implementation of such combined approaches. The toolkit currently contains layers for the DFTB method and an interface to the GFN1-xTB Hamiltonian, but due to its modular structure and its well-defined interfaces, additional atom-based schemes can be implemented easily. We are discussing the general structure of the framework, some essential implementation details, and several proof-of-concept applications demonstrating the perspectives of the combined methods and the functionality of the toolkit.

I. INTRODUCTION

Many recent technological advancements can be attributed, at least in part, to the discovery and development of novel functional materials. Computational simulation methods have become an important tool in the field of materials research in which the ability to explore configurational space and to accurately predict material properties is of paramount importance. *Ab initio* methods, such as Density Functional Theory (DFT), are commonplace as they offer a comparably high degree of accuracy and provide access to the electronic structure. However, their high computational cost makes them ill-suited to tackle problems requiring large time and size scales, such as molecular overlayer formation. While classical force field methods are able to reach the required scales at a reasonable cost, they do not provide access to electronic structure and are commonly unable to capture the full complexity of the potential energy surface; particularly when metals are involved. Semi-empirical methods are commonly referred to as "bridging" methods due to their ability to reach size and time scales that are off-limits to traditional *ab initio* methods while retaining electronic structure. Unfortunately, the parameter sets upon which such methods commonly rely require a non-trivial degree of effort to create and might suffer from limited transferability for complex systems. However, by leveraging the power of machine learning (ML), one could augment existing bridging methods to improve their transferability, reduce the complexity of creating parameter sets, and even introduce reactivity.

The field of machine learning has experienced steady growth over the past two decades and has recently exploded in popularity due to the accessibility afforded by frameworks, such as TensorFlow[1] and PyTorch.[2] Machine learning has been used with great success to tackle problems in a variety of fields, with chemistry, physics, and materials science being no exception. Several groups demonstrated successful and promising approaches for combining various levels of materials simulation methods with machine learning, such as in Refs. 3–9. We will omit an in-depth discussion of machine learning and its successes within the field of materials science in this paper, as detailed reviews[10–12] on the subject already exist, and the purpose of this publication is not to demonstrate any novel methods but to provide an overview about the current capabilities of the Tight-Binding Machine Learning Toolkit (TBMaLT) software package.

Applications of machine learning to materials modeling can be loosely placed into one of three categories: (*i*) assistive, (*ii*) augmentative, and (*iii*) substitutive. Assistive implementations are those which do not directly modify the method to which they are applied but rather fulfill an ancillary role, such as aiding the creation of new parameters.[13,14] Those which directly modify or replace parts of or add new components to their parent method are deemed to be augmentative; some examples being those replacing traditional parameterizations with neural networks or those offering alternative routes to the chemical Hamiltonian.[13,15,16] Novel ML-based approaches that are not templated off of any standard simulation method are labeled as substitutive as they are used in place of traditional methods; methods that directly predict specific properties, or groups thereof, are also included in this category. While such categorization is entirely artificial and somewhat subjective, it does highlight the notion that many approaches are based on established methods. Thus, assistive and augmentative approaches could, in theory, make use of existing codes to reduce the work required to implement them. When working with non-ML-based augmentations, one may directly modify existing codes to add new functionality and methods. However, this is not always viable for ML-based augmentations.

While novel ML-augmented methods are an attractive prospect, they can require a disproportionately large degree of effort to fully implement when compared to their non-ML counterparts. This is because the majority of production-level simulation packages were never designed to accommodate machine learning, and thus lack associated features, such as batch operability and automatic gradients. Therefore, existing code cannot easily be reused to reduce the work required, and one cannot simply generate and test a new method without first having to create and test a framework within which the new method can be used. With the Tight-Binding Machine Learning Toolkit (TBMaLT),[17] we would like to introduce an ML-capable framework that offers tested and documented, ready-to-use components to facilitate the rapid prototyping and development of new tight-binding models. It is hoped that the presence of such a framework will not only reduce the work required to implement, validate, and share new models but also aid in reducing duplicated efforts.

To keep the scope of the project to a reasonable size, the decision was made to limit the focus of TBMaLT exclusively to tight-binding type methods. Tight-binding theory [such as in Density Functional Tight-Binding (DFTB)[18,19] or in Extended Tight-Binding (xTB)[20,21]] was chosen because it lends itself well to the ideas of machine learning as it tends to focus on speed and scalability without giving up access to the electronic band structure. It is important to note that the purpose of this body of work is not to present a novel tight-binding model but rather to highlight the existence and capabilities of the TBMaLT framework.

II. STRUCTURE AND DESIGN

The main driving force behind TBMaLT's development was the common need for a testbed within which novel, commonly machine learning-based, tight-binding methods could be implemented without having to repeatedly write and test all the associated boilerplate code. Such a framework could, in addition to helping to reduce a potential source of duplicated effort, be employed to reduce the complexity associated with parameterizing currently existing models by automating much of the fitting process. The primary design philosophy behind this project emphasizes flexibility and usability over raw computational efficiency. TBMaLT's intended use case deviates from traditional modeling packages in that it is intended to facilitate the rapid prototyping, development, and validation of new models rather than the production-level use of existing ones.

The TBMaLT framework was written in Python as it is a highly accessible language, which lends itself well to rapid prototyping, which is prolifically used in the field of machine learning.[22] Critical machine learning functionality, such as automated analytical gradients, was provided through the use of the PyTorch package.[2] PyTorch was selected over other common machine learning frameworks due to its ease of use, simple pythonic syntax, and flexible nature. TBMaLT has been written in a modular and plug-and-play manner. This reduces the complexity associated with implementing new models and keeps users from having to modify the source code to do so. This has the added benefit of allowing new components to be used beyond their initial scope as components from different models could be arbitrarily combined to create new ones.

TBMaLT supports batch operability through the use of packed padding, which involves expanding a set of n rank k arrays to a common size and concatenating them into a single rank $(k+1)$ array. The padding values are commonly set to zero or one, depending on usage. This allows for operations to be vectorized and optimized for graphics processing unit (GPU)/single-instruction multiple data (SIMD) execution. The low memory efficiency of this method can commonly be offset via sparse tensors. However, it should be noted that the undeveloped nature of sparse tensors in PyTorch, at the time of writing, prevents their current use. Lorentzian and conditional broadening were used to alleviate the gradient stability issues with performing eigendecompositions on systems with degenerate eigenspectra, such as an equilibrated CH_4 molecule.[23,24] Conversion from the generalized to the standard eigenvalue problem was achieved via Cholesky decomposition as it produced more stable gradients than Löwdin orthogonalization. Self-consistency cycles, such as the self-consistent-charge (SCC) cycle of DFTB and xTB, are then performed (if applicable to the method), during which systems are sequentially culled from the batch as they converge to prevent over-convergence.

The default behavior in TBMaLT is to track the gradients during the SCC cycles using PyTorch's autograd engine, resulting in accurate gradients with respect to the input features and accurate forces. All results presented in the main paper were obtained this way. Depending on the system sizes, however, the overhead caused by the gradient tracking might become substantial. Different strategies can be applied in TBMaLT to reduce this overhead. One might calculate the converged charges (at a given value of the input feature vector) for each system outside of the purview of PyTorch's autograd engine and use those charges as input when building the charge dependent Hamiltonians (with gradient tracking) and omit any further SCC cycles. Keeping those input charges constant for several epochs, as first implemented by Yaron and co-workers,[13] helps to reduce the computational complexity associated with the gradient computation. Nevertheless, the input charges must be updated regularly (e.g., every tenth epoch as in Ref. 13) to take their change due to the changes in the input feature vector into account. Alternatively, in systems where one expects the converged charges to have only a weak dependence on the input feature vector (e.g., in mostly homogeneous systems), one might also carry out the external SCC cycles without the autograd feature for each input feature vector. In contrast to the previous approach, this yields continuous error functions without jumps, but the obtained gradients will always deviate from the exact ones due to the missing derivative of the converged charges with respect to the input feature vector. If the deviation is small, the optimization might, nevertheless, converge to a solution close to the one obtained with exact gradients (as demonstrated for rattled bulk silicon structures in Fig. S1).

In a typical workflow, a base `Calculator` is selected and initialized by supplying it with the requisite `Feed` objects. The `Feed` objects are optimizable entities that are responsible for supplying the necessary inputs to a given calculation. The `Calculator` is then tasked with interrogating these feeds and using the resulting data to compute various properties as and when they are requested. During initialization, the `Calculator` may also be provided with calculator-specific settings, such as those that may pertain to charge mixing, finite temperature, and so on. Once initialized, the `Calculator` is given a target system, or batch thereof, and is then itself interrogated to retrieve the desired properties. These properties can then be compared to those computed via the reference method, and a loss can be calculated. The PyTorch autograd engine can then be invoked, and the resulting gradients are used to update the properties contained within some or all of the feeds. A schematic depiction of this process is provided for the SCC-DFTB case in Fig. 1. The method can be modified by selecting a different choice of `Calculator` or `Feed` objects. For example, the SCC-DFTB implementation displayed in Fig. 1 can be morphed into xTB by swapping out the Hamiltonian and overlap `Feed` objects. The modular nature of TBMaLT means that in many cases, the `Feed` objects can themselves be modified or otherwise augmented to change the nature of the method.

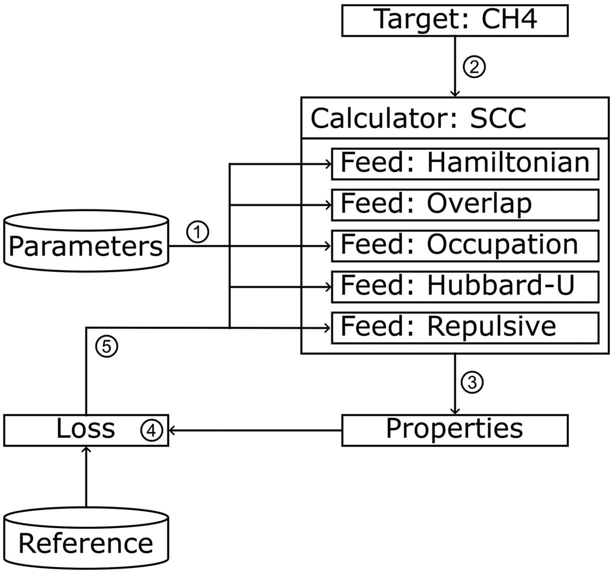


FIG. 1. Schematic depiction of the prediction and update process for an SCC-DFTB type calculator instance in which: (1) `Feed` objects are instantiated and supplied to the `Calculator`, (2) the target system is specified, (3) requested properties are predicted and then (4) compared to their reference values to get the loss upon which the autograd engine is then invoked, and (5) the feed properties are updated. Commonly steps 3–5 are repeated cyclically to improve the fit.

III. METHODOLOGY

The TBMaLT framework currently offers two different tight-binding approaches: the DFTB and the xTB methods. In the following, we give a concise summary of these two formalisms in order to supply the necessary theoretical background for the applications described in Sec. IV.

A. DFTB

The density functional tight-binding method[18,19] is an approximation to density functional theory (for a review, see Ref. 25). It starts from the Kohn–Sham ansatz, where the electron density $\rho(\mathbf{r})$ is represented as a sum of independent particle-densities,

$$\rho(\mathbf{r}) = \sum_i f_i |\psi_i(\mathbf{r})|^2, \tag{1}$$

where the sum runs over all one-electron states and f_i is the occupation number of state ψ_i. Writing the density as a deviation with respect to a reference density, $\rho(\mathbf{r}) = \rho_0(\mathbf{r}) + \delta\rho(\mathbf{r})$, the total energy is expanded up to second (or sometimes third) order and is traditionally collected into three energy terms,

$$E_{\mathrm{tot}}[\rho_0 + \delta\rho] = E_{\mathrm{rep}}[\rho_0] + E_{\mathrm{band}}[\rho_0] + E_2[\rho_0, \delta\rho^2], \tag{2}$$

the repulsive energy E_{rep}, the band energy E_{band}, and the charge transfer related second order energy E_2. The reference density is composed as a sum of the (compressed) atomic densities $\rho_0(\mathbf{r}) = \sum_A \rho_0^A(\mathbf{r})$ from all atoms A in the system. The last two terms in

Eq. (2) are explicitly calculated with several approximations, while the repulsive energy is used to compensate for the accuracy loss due to those approximations.

In the traditional DFTB approach, the repulsive energy is written as a sum of pairwise interactions,

$$E_{\text{rep}} = \frac{1}{2}\sum_{A,B\neq A} E_{\text{rep}}^{\text{sp}(A)\ \text{sp}(B)}(R_{AB}), \tag{3}$$

where the sum runs over all atom pairs in the system, and the two-body contributions $E_{\text{rep}}^{\text{sp}(A)\ \text{sp}(B)}$ depend on the species of the two atoms A and B, respectively, and their distances R_{AB}. The two-body contributions are usually obtained by a fitting procedure, requiring DFTB total energies and/or forces to reproduce corresponding reference quantities (typically originating from *ab initio* calculations).

The band-structure energy can be written as

$$E_{\text{band}} = \sum_i f_i\Big\langle\psi_i\Big| -\frac{1}{2}\nabla^2 + V_{\text{eff}}\Big|\psi_i\Big\rangle, \tag{4}$$

where the one-electron Hamiltonian $H = -\frac{1}{2}\nabla^2 + V_{\text{eff}}$ contains the kinetic energy and the Kohn–Sham effective potential V_{eff} calculated at the reference density $\rho_0(\mathbf{r})$. The one-electron wavefunctions are expanded into a linear combination of atomic orbitals

$$\psi_i(\mathbf{r}) = \sum_\mu c_{i\mu}\phi_\mu(\mathbf{r}), \tag{5}$$

which leads to the generalized eigenvalue problem

$$\sum_\nu H_{\mu\nu}c_{i\nu} = \varepsilon_i\sum_\nu S_{\mu\nu}c_{i\nu}, \tag{6}$$

with $H_{\mu\nu} = \langle\phi_\mu|H|\phi_\nu\rangle$ and $S_{\mu\nu} = \langle\phi_\mu|\phi_\nu\rangle$ being the Hamiltonian and overlap matrices, respectively, in the basis of the atomic orbitals.

The Hamiltonian matrix element $H_{\mu_A\nu_B}$ between the atomic orbitals ϕ_μ on atom A and ϕ_ν on atom B is approximated as

$$H_{\mu_A\nu_B} = \begin{cases} \varepsilon_\mu\delta_{\mu\nu} & \text{if } A = B,\\ \langle\phi_\mu| -\frac{1}{2}\nabla^2 + V_{\text{eff}}[\rho_A^0 + \rho_B^0]|\phi_\nu\rangle & \text{if } A \neq B, \end{cases} \tag{7}$$

where ε_μ is the electron eigenenergy (onsite energy) belonging to orbital ϕ_μ in a free atom. Due to the two-center approximation in the matrix elements between different atoms ($A \neq B$), the actual Hamiltonian (and overlap) matrices can be instantaneously constructed from pre-computed distance-dependent integrals for specific orbital orientations (ss^σ, sp^σ, pp^σ, pp^π, etc.) by applying the Slater–Koster transformations.[26]

One typically uses a minimal basis with maximal one polarization orbital in DFTB. In order to improve the accuracy, the basis functions and the atomic densities are obtained by solving the atomic Kohn–Sham equation for an atom in a confinement potential V_{conf},

$$\left[-\frac{1}{2}\nabla^2 + V_{\text{eff}}[\rho_I^0] + V_{\text{conf}}\right]\phi_\lambda = \tilde{\varepsilon}_\lambda\phi_\lambda. \tag{8}$$

Often, the confinement potential is chosen to be a harmonic potential $V_{\text{conf}}(r) = (r/r_0)^2$ and different values for the compression radius r_0 are used when calculating the compressed atomic orbitals ϕ_λ and the compressed atomic density ρ_I^0 (in separate calculations) for a given atom type I.

Finally, the charge fluctuation-dependent energy term is written as a Coulomb-like interaction of the gross Mulliken monopole charges on the atoms

$$E_2 = \frac{1}{2}\sum_{A,B}\gamma_{AB}\,q_A\,q_B, \tag{9}$$

where the coupling

$$\gamma_{AB} = \frac{1}{R_{AB}} - F(U_A, U_B, R_{AB}) \tag{10}$$

between the charges q_A and q_B on atoms A and B depends on the distance R_{AB} between the atoms and on the second derivative of the energy of the free DFT atom with respect to the charge, $U_I = \partial^2 E_I^{\text{atom}}/\partial q_I^2$ (chemical hardness). The second term in Eq. (10) is short-ranged. The Coulomb term is long-ranged, though, and needs special summation techniques (e.g., Ewald summation[27]) when calculated for periodic systems. The charge fluctuations also contribute to the Hamiltonian as

$$H_{\mu_A\nu_B}^2 = \frac{1}{2}S_{\mu\nu}\sum_C(\gamma_{AC} + \gamma_{BC})q_C \tag{11}$$

and consequently also to the resulting electronic structure. Since the Mulliken population depends on the solution of Eq. (6), a self-consistent procedure is needed to obtain the final quantities of a calculation, similar to the self-consistent field (SCF) in DFT calculations.

B. xTB

The extended tight-binding (xTB) method is rooted in the same energy expansion as DFTB but contains different energy expressions and approximations and employs a mostly atom-wise parameterization strategy. Note that the following expressions are restricted to the first iteration of the xTB method (GFN1-xTB) as TBMaLT currently implements the GFN1-xTB Hamiltonian.

The GFN1-xTB total energy reads as

$$E_{\text{tot}} = E_{\text{rep}} + E_{\text{disp}} + E_{\text{EHT}} + E_{\text{el}}^2 + E_{\text{el}}^3 + E_{\text{XB}} + G_{\text{Fermi}}, \tag{12}$$

whereby the zeroth order energy expansion is given by a repulsion term E_{rep} approximated by an atom-pairwise screened Coulomb potential,[28] and a dispersion term E_{disp} given by the established DFT-D3(BJ) dispersion correction scheme.[29,30] For the first order energy, an extended Hückel-type (EHT) term E_{EHT} is employed,

$$E_{\text{EHT}} = \frac{1}{2}\sum_{\mu\nu}P_{\nu\mu}H_{\mu\nu}^0, \tag{13}$$

where H^0 and P denote the core Hamiltonian and the density matrix, respectively. This energy term is the analog of E_{band} in Eq. (2).

Interatomic electrostatic interactions E_{el} originate from the second and third order energy expansion and are expressed with orbital-resolved Mulliken charges q_μ similar to DFTB. In fact, E^2_{el} closely resembles Eq. (9),

$$E^2_{\text{el}} = \frac{1}{2}\sum_{\mu,\nu} q_\mu q_\nu J_{\mu\nu}\,, \tag{14}$$

replacing the Coulomb interaction with the analytical Mataga–Nishimoto–Ohno–Klopman kernel[31–33]

$$J_{\mu_A,\nu_B} = \left(R^g_{AB} + f_{\text{avg}}(U_{A,l}, U_{B,l'})^{-g}\right)^{-1/g}, \tag{15}$$

where f_{avg} denotes the (harmonic) average of the atomic Hubbard parameters $U_{A,l}$ and $U_{B,l'}$, and g is a free parameter. The third order term E^3_{el} includes the diagonal elements of the charge derivative of the Hubbard parameter Γ_μ,

$$E^3_{\text{el}} = \frac{1}{3}\left(\sum_\mu \Gamma_\mu q_\mu\right)^3, \tag{16}$$

improving the description of highly charged systems.[34] A classical halogen bond correction E_{XB} is added to remedy the poor description of halogen bonds due to the isotropic point charge approximation. Finally, the electronic free energy G_{Fermi} is added.[35] This term arises from fractional occupations generated by Fermi smearing in order to handle static correlation cases.

Variational minimization of the total energy expression yields the tight-binding Hamiltonian matrix elements $H_{\mu\nu}$ for the generalized eigenvalue problem [Eq. (6)]. The Hamiltonian can be separated in a core Hamiltonian H^0 stemming from E_{EHT} and a charge-dependent Hamiltonian H^1 derived from E_{el}: $H = H^0 + H^1$.

Similar to DFTB, the extended tight-binding (xTB) Hamiltonian is defined in a valence-only but partially polarized minimal basis set of spherical Gaussian-type orbitals (GTO). The GTO contraction coefficients are obtained by expanding Slater-type orbitals (STO) using the STO-nG scheme.[36] The core Hamiltonian matrix elements are obtained in an EHT-like fashion by

$$H^0_{\mu\nu} = \frac{H_{\mu\mu} + H_{\nu\nu}}{2}\, S_{\mu\nu}\, X_{\text{EN}}\, F_{\text{scale}}\,, \tag{17}$$

where $S_{\mu\nu}$ denotes the overlap matrix elements, X_{EN} an electronegativity-dependent term, and F_{scale} an elaborate scaling function. The latter depends on the angular momenta of the shells and the distance of the atomic centers and consumes the majority of the xTB parameters. The diagonal elements $H_{\mu\mu/\nu\nu}$ are computed from parameterized, shell-resolved atomic level energies h_μ and their local environment described by the coordination number CN_μ as

$$H_{\mu\mu} = h_\mu + k^{\text{CN}}_\mu \text{CN}_\mu \tag{18}$$

with the scaling factor k^{CN}_μ being an adjustable parameter.

The charge-dependent Hamiltonian takes a similar form as DFTB's charge fluctuation Hamiltonian [Eq. (11)] but extends it by the third order term

$$H^1_{\mu\nu} = -\frac{1}{2} S_{\mu\nu}\left(\sum_\kappa (J_{\mu\kappa} + J_{\nu\kappa}) q_\kappa + \left(\Gamma_\mu q^2_\mu + \Gamma_\nu q^2_\nu\right)\right). \tag{19}$$

Correspondingly, and similar to DFTB, xTB also requires a self-consistent procedure.

The original implementation of xTB, including a more detailed description of the energy expressions, can be found in Refs. 20 and 21.

IV. RESULTS

In this section, we present the first proof-of-concept showcases demonstrating the current capabilities of the TBMaLT framework. We include both, DFTB and xTB applications, in order to emphasize the Hamiltonian-type agnostic structure of the framework. Since the applications described below are meant to serve as simple showcases only, we have chosen rather small training set sizes (which we have found to be suitable for those cases[37]), but omitted a detailed systematic study on the influence of the training set choice on the final results.

A. DFTB

1. Global and local parameter optimization

The traditional parameterization technique in DFTB involves an initial choice of atomic orbitals obtained from confined atomic calculations, which are then used to calculate the two-center approximated Hamiltonian integrals and the overlap integrals, as shown in Eqs. (5), (7), and (8). Within the TBMaLT framework, we have tested three different optimization strategies that have the potential to improve this procedure.

In the first approach, we carry out a global optimization of the compression radii of the species-dependent confinement potentials [V_{conf} in Eq. (8)] and the onsite energies [ε_μ in Eq. (7)]. This way, we obtain a "global" basis set for every species adapted to the training set. This approach is similar to the traditional parameterization approach but allows for an optimized choice of the basis orbitals and the onsite energies.

The second approach is similar, but we represent the distance-dependent two-center integrals of Eq. (7) and the similar overlap integrals with splines and optimize the spline coefficients directly. This method is similar to the approach followed by Yaron and co-workers[13] and yields globally optimized integral tables for the corresponding species pairs. Unfortunately, by optimizing the two-center diatomic integrals directly, one loses the concept of a well-defined basis. This inevitably leads to inconsistencies between the obtained Hamiltonian and overlap matrices, although those can be limited by starting the optimization from a good and consistent initial guess. Furthermore, one cannot represent calculated quantities (such as electron density or eigenstates) as a function of space anymore.

In our third approach, we reintroduce the concept of the well-defined basis by predicting individual confinement radii V_{conf} and onsite energies ε_μ for each atom based on its local environment. This way, the basis becomes atom-dependent instead of atom-type dependent, as in the traditional approach. The Hamiltonian and the overlap matrices can be consistently calculated within this basis, and all obtained quantities have a well-defined real space representation.

In all three approaches, the optimization is driven by using the loss function

$$F = \frac{1}{N}\sum_{j=1}^{m}\sum_{i=1}^{N}\omega_j\big(P_{ij}^{\mathrm{DFT}} - P_{ij}^{\mathrm{DFTB}}\big)^2, \tag{20}$$

where N describes the number of systems in the training dataset, and m is the number of physical properties taken into account. P^{DFT} and P^{DFTB} are the target physical property values from the reference DFT and the DFTB calculations, respectively, and ω_j is the weight associated with a given physical property.

For the training procedure, we used a subset of the ANAKIN-ME dataset,[38] where every molecule contained one heavy atom only (ANI-1_1). From this subset, we have randomly selected 1000 and 400 molecules for the training and test sets, respectively, for all three optimization methods. In order to reduce the bias in our training, we have carried out three independent training runs for each optimization method, selecting the molecules of the training and test sets randomly from the ANI-1_1 dataset each time. The reference DFT data were calculated using the FHI-aims code.[39] We have used the Perdew–Burke–Ernzerhof (PBE) functional[40] with the "tight" basis set. For the local basis optimization, we have used atom-centered symmetry functions (ACSFs)[41] to represent the chemical environment of the atoms in the ML input and the random forest method as the ML algorithm.[42] The ACSFs have been implemented in TBMaLT, while Scikit-learn[43] has been used to perform the random forest predictions. (Further details about the ML settings can be found in the supplementary material.)

Figure 2 shows the result of the optimizations when using the dipole moment as the only target property in the loss function ($m = 1$). The reported mean absolute errors (MAEs) are the averages of the MAEs calculated over the test sets in each of the three training runs. For comparison, we also report the performance when using the `mio-1-1` parameter set.[19] All three optimization techniques show a significant improvement, with the local basis optimization model performing best.

A more detailed discussion of the three different optimization techniques and further use cases can be found in Ref. 37.

2. Training on the density of states for periodic systems

The TBMaLT framework can also treat periodic systems and allows for their inclusion into optimization and training scenarios (currently with the DFTB Hamiltonian only). This should be demonstrated by the following showcase, where DFTB parameters were trained on bulk silicon structures in order to improve the density of states (DOS) description. Unlike discrete properties, such as Mulliken charges or dipole moments, the DOS is represented by a continuous distribution in a certain energy range. Therefore, the Hellinger distance[44] was employed as a loss function to evaluate the difference between pairs of distributions as it enables learning not only on the absolute differences but also on the trends of the curves. By using a similar strategy as discussed in Sec. IV A 1, the diatomic two-center integrals were represented by splines, which were optimized in order to obtain the Hamiltonian and the overlap matrices yielding an improved DOS for the training dataset.

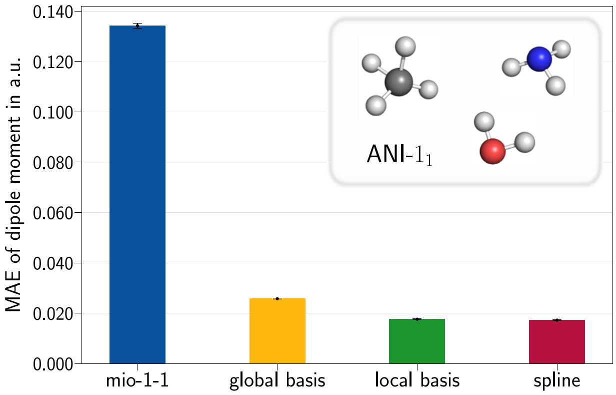


FIG. 2. Mean absolute error of dipole moments from DFTB calculations with respect to reference DFT calculations. The Hamiltonian and overlap matrices of the DFTB calculations have been constructed using the `mio − 1 − 1` parameter set (blue), the globally optimized basis model (yellow), the environment dependently optimized basis model (green), and the globally optimized splines of the diatomic two-center integrals (red). The values with the corresponding error bars represent averages over the test sets of three independent training runs, with the training and test set sizes of 1000 and 400, respectively.

First, we generated a database containing 50 randomly rattled silicon bulk systems with supercells of 64 atoms [an example is shown in Fig. 3(a)]. DFT calculations employing the HSE06 hybrid functional[45] with a screening parameter of 0.11 a_0^{-1} were carried out with the FHI-aims code[39] using the "tight" parameter set. During the training process, the `siband-1-1` parameter set[46] was used as the starting point for the optimization of the spline representations of the DFTB diatomic Hamiltonian and overlap integrals, which were optimized simultaneously. In order to ensure that the DFT and DFTB DOS results are comparable, the DOS was calculated by applying identical Gaussian broadening to the eigenvalues in both cases. The same energy ranges (from −3.3 eV to +1.6 eV relative to the Fermi levels) were used to sample the DOS curves and to calculate the loss between the DFT and the DFTB results. To reduce the influence of sampling on the training data, 30 systems were randomly selected and partitioned into three distinct training sets, yielding three independent training runs. The remaining 20 unseen systems were then used to test the model. Figures 3(b) and 3(c) compare the DOS results with the DFT references on the test dataset in the three training runs before and after the ML optimization of the DFTB parameters. The ML optimization leads to more accurate results, reducing the average Hellinger and MSE losses from 10.8 and 19.4 to 6.2 and 5.4, respectively. It is also clear that the results after the ML optimization fit the DFT references better than the original DFTB calculations using the `siband-1-1` parameter set.

Additional tests were also performed to evaluate the transferability of the model; i.e., whether it makes reasonable predictions for larger systems after having been trained on small ones. This was done by training models exclusively on the small 64-atom systems and evaluating them on larger 512-atom systems. In order to reduce the computational efforts, training and testing references were obtained by using the GGA-PBE functional[40] in this case. As shown in Fig. 3(d), the ML optimization can correct the energy

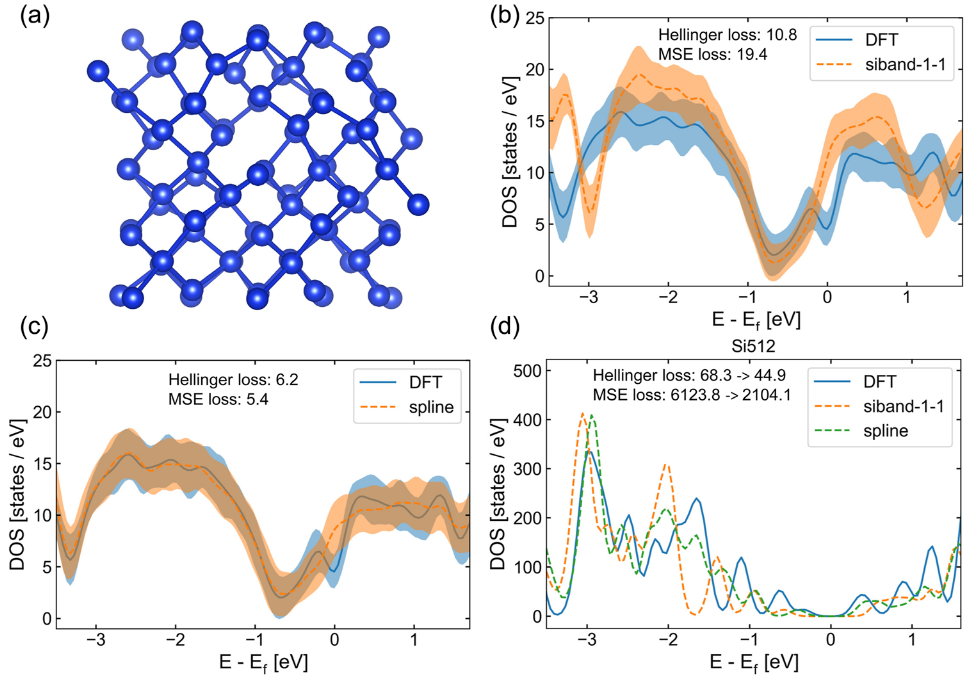


FIG. 3. Examples of DOS training: (a) geometry of a rattled diamond type silicon system, (b) comparison of the DFTB DOS curves obtained with three test runs using the `siband-1-1` parameter set with the DFT references, (c) comparison of the DFTB DOS curves obtained with three test runs with optimized splines for the diatomic integrals with the DFT references, and (d) DOS curves of the transferability test calculated by DFT, by DFTB with the `siband-1-1` parameter set, and by DFTB with optimized splines for the diatomic integrals. In (b) and (c), the solid blue and dashed orange lines represent the average DOS values obtained with three test runs with DFT and DFTB, respectively, while the shadow areas indicate the diversity of the corresponding DOS values.

shift of the DOS curve below the Fermi level and enhance the accuracy above the Fermi level, making a total improvement of 34% and 66% as measured by the Hellinger loss and the MSE loss, respectively.

B. xTB

This section illustrates the possibility of using other tight-binding schemes within the TBMaLT framework. For this purpose, we employ the GFN1-xTB Hamiltonian to improve its parameterization for applications on specific chemical spaces. The Hamiltonian is integrated into the TBMaLT workflow employing a GFN1-xTB adapter.[47] Prior to further analysis, it is verified that the xTB Hamiltonian reproduces the number of electrons as well as the orbital occupancies within TBMaLT.

The optimization procedure employs gradient descent with the Adam optimizer[48] and an MSE loss function. The atomic charges obtained from SCC-DFTB calculations are used as the target quantity. The corresponding DFT reference charges are obtained as described in Sec. IV A. Using the ANAKIN-ME subset ANI-1_1 with a single heavy atom (C, N, and O), Hamiltonian-dependent parameters (scaling of electronegativities k_{EN} and shell parameters) as well as element-specific scaling parameters (elemental pair parameters k_{pair}) are optimized. For training purposes, we utilize 50 samples and test on 1000 samples, both containing an equal ratio of molecular species. Despite the minimal split, the chemical space of the dataset is well represented due to the choice of only a single heavy atom. Moreover, it facilitates only minor deviations from the original general-purpose GFN1-xTB parameterization and demonstrates the general robustness of the fit, being able to transfer to a multitude of samples not included in the fit. However, since the study is intended as a proof-of-principle, one should notice that the chemical variety in the employed dataset is rather limited.

Figure 4 shows the performance (MAE) of the default and various reoptimized GFN1-xTB parameterizations with respect to the DFT reference values on the ANI-1_1 dataset. For training

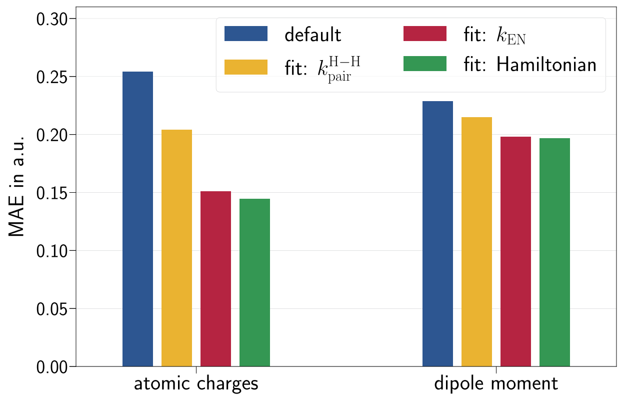


FIG. 4. Comparison of reparameterizations of different parameters on ANI-1_1 data. Hereby, solely atomic charges are utilized as the fit target. Furthermore, the change in nominal value of dipole moments for the new parameterizations is given to demonstrate the transferability of our approach. The deviation is given by the MAE to the reference method. The optimized parameters comprise the element pair-specific scaling parameter $k_{\mathrm{pair}}^{\mathrm{H-H}}$ (yellow), the electronegativity scaling parameter k_{EN} (red), as well as all global Hamiltonian-related parameters of GFN1-xTB including k_{EN} (green). In particular, the reparameterization based on electronegativities k_{EN} shows a significant improvement over the existing default parameterization.

on the atomic charges (Fig. 4, left), reparameterizations based on Hamiltonian- (red, green) and element-specific (yellow) parameters lead to a significant improvement over the initial parameterization (blue). As refitting all Hamiltonian-based parameters grants the most flexibility to the optimization, the resulting reparameterization expectedly achieves the largest improvement of ~44%. Considering the only slightly worse performance (~40% error decrease) of the single-parameter fit of the electronegativity scaling k_{EN} (red), however, it becomes apparent that this parameter is most important and mainly drives the Hamiltonian fit. This is due to the fact that the original parameterization is designed to give reliable results over a wide range of chemical space. Since this study targets small molecules with limited binding motifs and simple structures, the electronic properties are optimized for individual heavy atoms.

Furthermore, it becomes evident that the reparameterization based on the atomic charges also has a beneficial effect on other properties, as exemplified by the dipole moments (Fig. 4, right). This demonstrates the possibility of improving the overall accuracy for specific use cases by means of a GFN1-xTB reparameterization.

V. SUMMARY

Combining simple tight-binding models with machine learning is a very promising approach for materials simulations. Machine learning can improve the accuracy of such models significantly while having a well-defined physical model provides the advantage of an easier interpretation of the simulation results as compared to approaches, where physical quantities are predicted "directly" via machine learning without an underlying classical or quantum mechanical model. Also, the training procedure can be expected to be less involved when the physics is provided by the tight-binding model. Derivation of the ground state and excited state quantities not considered during training is straightforward in these approaches, and their transferability should be considerably better as of the direct approaches, provided the physical behavior is qualitatively well described by the chosen model.

We have started the development of the open-source software package TBMaLT,[17] which provides a toolkit for combining machine learning approaches with atomistic simulation models. Our main goal is to provide a framework for the rapid prototyping of such hybrid approaches and for testing their capabilities and limits. The toolkit has been written using the PyTorch framework, providing backpropagated gradients out of the box. TBMaLT's overall structure is designed to be physical model agnostic, opening the possibility for implementing and testing arbitrary (atom-based) approaches. Although the development only started recently, the package already offers the DFTB model (as a built-in module) and the xTB model (via a connector to an external PyTorch-based xTB implementation). Further models can be implemented or integrated easily. TBMaLT comes with extensive unit tests and a well-documented application programming interface.

We have presented a few proof-of-concept showcases, which demonstrate the viability of such hybrid approaches as well as the current capabilities of the TBMaLT framework. Our project's development follows the usual open-source development workflow and is open to external contributors. We hope that the software package will be useful for the scientific community and will help to catalyze the development of faster and more accurate materials simulation methods in the near future.

SUPPLEMENTARY MATERIAL

The supplementary material contains further details to the comparison of the approximative external SCC cycle technique with the exact gradient calculation (as discussed in Sec. II) for a rattled silicon structure and additional details about the machine learning settings used in Sec. IV A 1. Additionally, a dataset containing the geometries and the *ab initio* reference data used for the show case applications in Sec. IV is provided.

ACKNOWLEDGMENTS

This work was supported by the DFG Priority Program No. SPP 2363 on "Utilization and Development of Machine Learning for Molecular Applications—Molecular Machine Learning" (Project No. 497190956). The authors also gratefully acknowledge funding from the DFG Research Training Network GRK 2247 on "Quantum Mechanical Materials Modeling—QM^3."

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

A. McSloy: Conceptualization (equal); Methodology (equal); Software (equal); Supervision (equal); Writing – original draft (equal). **G. Fan**: Conceptualization (equal); Software (equal); Writing – original draft (equal). **W. Sun**: Conceptualization (equal); Software (equal); Writing – original draft (equal). **C. Hölzer**: Conceptualization (equal); Software (equal); Writing – original draft (equal). **M. Friede**: Conceptualization (equal); Software (equal); Writing – original draft (equal). **S. Ehlert**: Conceptualization (equal); Software (equal); Writing – original draft (equal). **N.-E. Schütte**: Software (equal). **S. Grimme**: Funding acquisition (equal). **T. Frauenheim**: Funding acquisition (equal). **B. Aradi**: Conceptualization (equal); Project administration (equal); Software (equal); Supervision (equal); Writing – original draft (equal).

DATA AVAILABILITY

The TBMaLT software package can be downloaded from its GitHub repository.[17] Its API is extensively documented in the source code using Python's docstring feature. Geometries and the target quantities obtained from DFT reference calculations for the various training and test tests are provided in the supplementary material.

REFERENCES

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden,

M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," arXiv:1603.04467 (2016).

[2] Pytorch: An imperative style, high-performance deep learning library, 2019.

[3] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. von Lilienfeld, "Big data meets quantum chemistry approximations: The delta-machine learning approach," J. Chem. Theory Comput. **11**, 2087–2096 (2015).

[4] Z. Qiao, M. Welborn, A. Anandkumar, F. R. Manby, and T. F. Miller, "OrbNet: Deep learning for quantum chemistry using symmetry-adapted atomic-orbital features," J. Chem. Phys. **153**, 124111 (2020).

[5] G. Zhou, B. Nebgen, N. Lubbers, W. Malone, A. M. N. Niklasson, and S. Tretiak, "Graphics processing unit-accelerated semiempirical Born Oppenheimer molecular dynamics using PyTorch," J. Chem. Theory Comput. **16**, 4951–4962 (2020).

[6] S. Wengert, G. Csányi, K. Reuter, and J. T. Margraf, "Data-efficient machine learning for molecular crystal structure prediction," Chem. Sci. **12**, 4536–4546 (2021).

[7] S. Dick and M. Fernandez-Serra, "Highly accurate and constrained density functional obtained with differentiable programming," Phys. Rev. B **104**, L161109 (2021).

[8] T. Zubatiuk, B. Nebgen, N. Lubbers, J. S. Smith, R. Zubatyuk, G. Zhou, C. Koh, K. Barros, O. Isayev, and S. Tretiak, "Machine learned Hückel theory: Interfacing physics and deep neural networks," J. Chem. Phys. **154**, 244108 (2021).

[9] X. Zhang and G. K.-L. Chan, "Differentiable quantum chemistry with PySCF for molecules and materials at the mean-field level and beyond," J. Chem. Phys. **157**, 204801 (2022).

[10] J. Schmidt, M. R. G. Marques, S. Botti, and M. A. L. Marques, "Recent advances and applications of machine learning in solid-state materials science," npj Comput. Mater. **5**, 83 (2019).

[11] K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev, and A. Walsh, "Machine learning for molecular and materials science," Nature **559**, 547–555 (2018).

[12] G. Pilania, "Machine learning in materials science: From explainable predictions to autonomous design," Comput. Mater. Sci. **193**, 110360 (2021).

[13] H. Li, C. Collins, M. Tanha, G. J. Gordon, and D. J. Yaron, "A density functional tight binding layer for deep learning of chemical Hamiltonians," J. Chem. Theory Comput. **14**, 5764–5776 (2018).

[14] H. Liu, Y. Li, Z. Fu, K. Li, and M. Bauchy, "Exploring the landscape of Buckingham potentials for silica by machine learning: Soft vs hard interatomic forcefields," J. Chem. Phys. **152**, 051101 (2020).

[15] C. Schattauer, M. Todorović, K. Ghosh, P. Rinke, and F. Libisch, "Machine learning sparse tight-binding parameters for defects," npj Comput. Mater. **8**, 116 (2022).

[16] L. Zhang, B. Onat, G. Dusson, A. McSloy, G. Anand, R. J. Maurer, C. Ortner, and J. R. Kermode, "Equivariant analytical mapping of first principles Hamiltonians to accurate and transferable materials models," npj Comput. Mater. **8**, 158 (2022).

[17] TBMaLT: Tight Binding Machine Learning Toolkit, https://github.com/tbmalt/tbmalt; accessed: 30 October 2022.

[18] G. Seifert, D. Porezag, and T. Frauenheim, "Calculations of molecules, clusters, and solids with a simplified LCAO-DFT-LDA scheme," Int. J. Quantum Chem. **58**, 185 (1996).

[19] M. Elstner, D. Porezag, G. Jungnickel, J. Elsner, M. Haugk, T. Frauenheim, S. Suhai, and G. Seifert, "Self-consistent-charge density-functional tight-binding method for simulations of complex materials properties," Phys. Rev. B **58**, 7260 (1998).

[20] S. Grimme, C. Bannwarth, and P. Shushkov, "A robust and accurate tight-binding quantum chemical method for structures, vibrational frequencies, and noncovalent interactions of large molecular systems parametrized for all spd-block elements (Z = 1–86)," J. Chem. Theory Comput. **13**, 1989–2009 (2017).

[21] C. Bannwarth, E. Caldeweyher, S. Ehlert, A. Hansen, P. Pracht, J. Seibert, S. Spicher, and S. Grimme, "Extended tight-binding quantum chemistry methods," WIREs Comput. Mol. Sci. **11**, e1493 (2021).

[22] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual* (CreateSpace, Scotts Valley CA, 2009).

[23] M. Seeger, A. Hetzel, Z. Dai, E. Meissner, and N. D. Lawrence, "Auto-differentiating linear algebra," arXiv:1710.08717 [cs.MS] (2017).

[24] H. J. Liao, J. G. Liu, L. Wang, and T. Xiang, "Differentiable programming tensor networks," Phys. Rev. X **9**, 031041 (2019).

[25] M. Elstner and G. Seifert, "Density functional tight binding," Philos. Trans. R. Soc., A **372**, 20120483 (2012).

[26] J. C. Slater and G. F. Koster, "Simplified LCAO method for the periodic potential problem," Phys. Rev. **94**, 1498 (1954).

[27] P. P. Ewald, "Die berechnung optischer und elektrostatischer gitterpotentiale," Ann. Phys. **369**, 253–287 (1921).

[28] S. Grimme, "A general quantum mechanically derived force field (QMDFF) for molecules and condensed phase simulations," J. Chem. Theory Comput. **10**, 4497–4514 (2014).

[29] S. Grimme, J. Antony, S. Ehrlich, and H. Krieg, "A consistent and accurate *ab initio* parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu," J. Chem. Phys. **132**, 154104 (2010).

[30] E. R. Johnson and A. D. Becke, "A post-Hartree–Fock model of intermolecular interactions," J. Chem. Phys. **123**, 024101 (2005).

[31] K. Nishimoto and N. Mataga, "Electronic structure and spectra of some nitrogen heterocycles," Z. Phys. Chem. **12**, 335–338.

[32] K. Ohno, "Some remarks on the Pariser-Parr-Pople method," Theor. Chim. Acta **2**, 219–227.

[33] G. Klopman, "A semiempirical treatment of molecular structures. II. Molecular terms and application to diatomic molecules," J. Am. Chem. Soc. **86**, 4550–4557.

[34] Y. Yang, H. Yu, D. York, Q. Cui, and M. Elstner, "Extension of the self-consistent-charge density-functional tight-binding method: Third-order expansion of the density functional theory total energy and introduction of a modified effective Coulomb interaction," J. Phys. Chem. A **111**, 10861–10873 (2007).

[35] N. D. Mermin, "Thermal properties of the inhomogeneous electron gas," Phys. Rev. **137**, A1441–A1443.

[36] W. J. Hehre, R. F. Stewart, and J. A. Pople, "Self-consistent molecular-orbital methods. I. Use of Gaussian expansions of slater-type atomic orbitals," J. Chem. Phys. **51**, 2657–2664 (1969).

[37] G. Fan, A. McSloy, B. Aradi, C.-Y. Yam, and T. Frauenheim, "Obtaining electronic properties of molecules through combining density functional tight binding with machine learning," J. Phys. Chem. Lett. **13**, 10132–10139 (2022).

[38] J. S. Smith, O. Isayev, and A. E. Roitberg, "ANI-1: An extensible neural network potential with DFT accuracy at force field computational cost," Chem. Sci. **8**, 3192–3203 (2017).

[39] V. Blum, R. Gehrke, F. Hanke, P. Havu, V. Havu, X. Ren, K. Reuter, and M. Scheffler, "*Ab initio* molecular simulations with numeric atom-centered orbitals," Comput. Phys. Commun. **180**, 2175–2196 (2009).

[40] J. P. Perdew, K. Burke, and M. Ernzerhof, "Generalized gradient approximation made simple," Phys. Rev. Lett. **77**, 3865 (1996).

[41] J. Behler, "Atom-centered symmetry functions for constructing high-dimensional neural network potentials," J. Chem. Phys. **134**, 074106 (2011).

[42] L. Breiman, "Random forests," Mach. Learn. **45**, 5–32 (2001).

[43] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," J. Mach. Learn. Res. **12**, 2825–2830 (2011).

[44] E. Hellinger, J. Reine Angew. Math. **1909**, 210–271.

[45] J. Heyd, G. E. Scuseria, and M. Ernzerhof, J. Chem. Phys. **118**, 8207–8215 (2003).

[46] S. Markov, B. Aradi, C.-Y. Yam, H. Xie, T. Frauenheim, and G. Chen, IEEE Trans. Electron Devices **62**, 696–704 (2015).

[47] Fully Differentiable Approach to Semiempirical Extended Tight-Binding, https://github.com/grimme-lab/dxtb. Accessed: 2022-10-28, Package will be open-sourced upon dxTB publication. Prior access can be granted upon request.

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv.1412.6980 [cs].