PROJECT SUMMARY / ABSTRACT

OpenMM [http://openmm.org] is the most widely-used open source GPU-accelerated framework for biomolecular modeling and simulation (>1300 citations, >270,000 downloads, >1M deployed instances). Its Python API makes it widely popular as both an application (for modelers) and a library (for developers), while its C/C++/Fortran bindings enable major legacy simulation packages to use OpenMM to provide high performance on modern hardware. OpenMM has been used for probing biological questions that leverage the $14B global investment in structural data from the PDB at multiple scales, from detailed studies of single disease proteins to superfamily-wide modeling studies and large-scale drug development efforts in industry and academia.

Originally developed with NIH funding by the Pande lab at Stanford, we aim to fully transition toward a community governance and sustainable development model and extend its capabilities to ensure OpenMM can power the next decade of biomolecular research. To fully exploit the revolution in QM-level accuracy with machine-learning (ML) potentials, we will add plug-in support for ML models augmented by GPU-accelerated kernels, enabling transformative science with QM-level accuracy. To enable high-productivity development of new ML models with training dataset sizes approaching 100 million molecules, we will develop a Python framework to enable OpenMM to be easily used within modern ML frameworks such as TensorFlow and PyTorch. Together with continued optimizations to exploit inexpensive GPUs, these advances will power a transformation within biomolecular modeling and simulation, much as deep learning has transformed computer vision.

SPECIFIC AIMS

Predictive, quantitative modeling of biomolecular systems is critical to understanding fundamental molecular mechanisms underlying biological function and disease, as well as developing small molecule and biological therapeutics. Two distinct approaches have emerged to model the chemical interactions in these simulations: **molecular mechanics (MM) models**, in which physical interactions are modeled at the atomic level, and **machine learning (ML) models**, in which a flexible function approximator fit to data makes quantitative predictions. While these approaches have traditionally been disjoint, recent work to bring these two together to develop **quantum machine learning (QML) potentials** has demonstrated the enormous potential for differentiable many-body potentials learned by deep neural networks to substitute for or supplement traditional physical force fields or expensive quantum chemical calculations. Despite their enormous promise to power a new decade of discovery in biomolecular modeling and simulations, a host of technical limitations stand in the way of practical widespread use.

Here, we propose to both sustain and significantly extend our popular **OpenMM Toolkit** to realize this goal. OpenMM is the most widely-used GPU-accelerated framework for biomolecular simulation (>1300 citations, >270,000 downloads, >1M deployed instances). First, we will transition OpenMM to a distributed development and governance model that will ensure it can continue to meet the needs of the biomolecular simulation community for years to come. Second, we add flexible support for plug-in machine learning models to enable multiple uses: (1) as quantum machine learning (QML) potentials able to substitute for traditional MM potentials while delivering extraordinary accuracy in new and existing applications; (2) as collective variables (CVs) or studying rare events or conformational changes; and (3) as advanced integrators or samplers for simulating long timescales or multiscale dynamics. These plug-ins will be hardware accelerated and fully supported by the existing OpenMM ecosystem, which includes numerous tools for enhanced sampling, protein-ligand free energy calculations, and rare event simulations. Third, to accelerate development of next-generation hybrid QML models that blend traditional QML approaches with physical long-range potentials, we will develop a library for making use of hardware-accelerated atomic features and OpenMM physical potential terms within popular machine learning frameworks such as TensorFlow and PyTorch. Together with continued optimizations to exploit emerging accelerated hardware architectures (TPUs, tensor cores, etc), these technological advances will enable and explore a variety of new scientific investigations that were previously computationally impossible and limited by force field accuracy.

Aim 1: Transition OpenMM from a single-laboratory single-developer-led code to a community governance and distributed developer model to ensure long-term sustainability. Originally developed in a single laboratory, OpenMM has grown so quickly and is integrated so deeply within the biomolecular simulation community that a distributed development model is necessary to ensure a sustainable future. A new OpenMM Consortium will ensure OpenMM continues to meet the most urgent needs fo the biomolecular modeling and simulation community.

Aim 2: Develop a next-generation hardware-accelerated molecular simulation platform that will enable direct interoperability with machine learning frameworks. We will extend OpenMM to allow facile integration of QML potentials and machine learning models created within popular frameworks such as TensorFlow, PyTorch, and JAX for use as new forces, collective variables, or integrators. In addition, we will develop custom hardware-accelerated kernels to enable these models to run at high speed within OpenMM by exploiting accelerator hardware such as GPUs, TPUs, and tensor cores. Driving applications include the use of hybrid physical/ML potentials to develop ultra-high accuracy protein-ligand free energy calculations generalizable to new chemistries and the study of enzyme mechanisms or covalent inhibition with enormous speed advantages over traditional QM/MM methods.

Aim 3: Create a software infrastructure to enable hybrid machine learning / physical model development. We will develop a library of hardware-accelerated kernels for QML atomic features and physical force terms to be easily used within popular machine learning ecosystems (such as TensorFlow and PyTorch) as custom operations ("ops").This will enable developers to rapidly develop new QML potentials and hybrid physical/ML models able to deliver quantum chemical accuracy at molecular mechanics speeds, as well as enable high-throughput machine learning for structure-enabled biomolecular datasets in a manner that exploits both physical and ML features.

Aim 4: Develop community repositories of large quantum chemical datasets and models to accelerate development and deployment of QML potentials. We will work with MolSSI to create an online QML model repository and standardized QML model specification to allow developers to easily share, update, and deploy their models to OpenMM and other codes. To accelerate the development of QML models with increased accuracy and greatly expanded coverage, we will use the planetary-scale Folding@home distributed computing framework (∼330,000 cores) to create large public quantum chemical datasets made available through the MolSSI QCArchive.

Together, these aims will ensure OpenMM will significantly advance the capabilities of predictive modeling and simulation of biomolecular systems and produce new insight into the modeling of small molecule therapeutics.

SIGNIFICANCE

Biomolecular modeling and simulation is a key technology for leveraging the $14B global investment in biomolecular structure data in the protein databank. Molecular mechanics (MM) force fields model biomolecular systems at the atomistic scale, allowing researchers to probe a variety of phenomena that involve energetics, forces, and dynamics of biomolecular systems [1–3]. A wide variety of methodologies have been developed for making use of molecular mechanics models, from simple methods that use energy minimization or molecular dynamics to advanced calculations that involve algorithms for enhanced or targeted sampling [1]. Biomolecular modeling and simulation has a wide variety of applications, from probing the molecular mechanism of biological function [2, 4] and the molecular underpinnings of disease [5–7] to the interpretation of experiments [8] to quantitative predictions that guide the development of new small molecule therapeutics [9–11] and biologics [12, 13]. Because these investigations only require access to now-ubiquitous computational resources to access microsecond timescales [14], biomolecular modeling has become a facile and indispensable tool in the biosciences.

OpenMM is a critical technology for the biomolecular simulation and modeling community. OpenMM [15–20] is a biomolecular modeling and simulation code that is both fast and flexible, enabling researchers to directly use it as a research tool for accessing state-of-the-art algorithms on essentially any computing hardware or to use it as a library to build more complex modeling tools that take advantage of these algorithms in new and advanced ways. OpenMM's key papers [15–20] have been cited over 1300 times[1], the OpenMM conda package alone has been downloaded over 270,000 times [23], and it has been deployed on more than one million GPUs [24].

While relatively young, OpenMM provides a number of critical modern features not present in other biomolecular modeling and simulation packages that endow it with numerous advantages compared to legacy biomolecular modeling and simulation packages for a number of important application domains:

- **Speed:** OpenMM was one of the first[2] molecular simulation packages to exploit inexpensive consumer-grade graphics processors (GPUs) [18], which brought a 100× increase in performance/price ratio over CPUs and drove a revolution in biomolecular simulations. OpenMM features both CUDA and OpenCL computational platform backends that allow it to achieve extremely high performance on a wide variety of inexpensive consumer-grade GPUs, a multithreaded CPU backend to enable it to run on ubiquitous CPUs, and a plugin architecture that speeds the development of new kernels to exploit new accelerated hardware.
- **Productivity:** OpenMM provides a simple, expressive API [26] that makes it incredibly easy to build new applications that make use of molecular modeling operations in Python, C++, C, or Fortran without the need to wrap command-line tools. By using an object-oriented model that provides useful abstractions for biomolecular modeling, biosciences researchers can achieve high productivity just as modern machine learning researchers can with modern frameworks such as Tensorflow and PyTorch.
- **Python friendly:** The Python API allows scientists to either use OpenMM directly or build applications or libraries in Python, such as the openmmtools library [27] (downloaded >100,000 times [28]). The enormous number of scientific libraries available for Python makes it easy to quickly build new, complex applications or research projects in Python. Educational tutorials can also exploit modern Jupyter notebooks and run in the cloud to make it easy for new researchers to get started.
- **Ubiquity:** OpenMM is conda-installable [23], allowing it to be easily used as a dependency in other tools that can be automatically installed without the user even knowing they are using OpenMM under the hood.
- **Modularity:** Developers and users can easily add new forces, barostats, thermostats, collective variables, and integrators without modifying the core code (and often in pure Python!). New features and kernels can easily be added via a simple plugin architecture that can be built and distributed separately.
- **Extensibility:** Research and experimentation with new potential forms is incredibly easy with Custom Forces, which enable researchers to write algebraic expressions that are automatically optimized, differentiated, and compiled into GPU-accelerated kernels.
- **Interoperability:** OpenMM is force field agnostic, supporting all major molecular mechanics (MM) models. It provides a Python layer to make it easy to import biomolecular systems from other major simulation packages (such as AMBER [29], CHARMM [21], NAMD [30], or GROMACS [31]).

[1] The number of OpenMM citations is likely an underestimate of works using OpenMM because multiple legacy codes—such as CHARMM [21] and TINKER-OpenMM [22]—make use of OpenMM "under the hood", so users may not even know they are using OpenMM.

[2] The other major GPU-accelerated code, ACEMD [25]—created by co-I Fabritiis—now uses OpenMM under the hood.

- **Legacy support:** OpenMM is the “webkit for biomolecular simulation," in that it is ubiquitously found powering molecular simulation packages—with C++, C, and Fortran APIs, OpenMM is used under the hood in major legacy codes (including CHARMM, Tinker, and ACEMD) to enable these packages to provide high performance on modern computing hardware like GPUs without a complete rewrite.
- **Robustness:** OpenMM drives the GPU simulation core in the Folding@home worldwide distributed computing project, and has run on all major commercially-available GPUs and **over a million unique computer systems**, making it the most widely-tested and widely-deployed biomolecular simulation package. It also powers gpugrid.net through ACEMD, a biomolecular simulation package for drug discovery.
- **Simplicity:** With an extensive user guide, clear API docs, a simple script generator, and an entire ecosystem of tools built around it, OpenMM makes it easy for new users to dive in.
- **Free, open source, and permissively licensed:** Unlike many legacy biomolecular simulation packages, OpenMM is available free of charge and without the need to sign a license agreement. OpenMM is licensed under the permissive MIT License (with some of the CUDA and OpenCL components under the LGPL License) to enable it to have maximum impact in the biosciences, and allowing it to be used "under the hood" within other software applications, often without the user even knowing OpenMM is the underlying simulation engine. The MIT License allows OpenMM to be used, modified, and redistributed without restriction, provided attribution is maintained, ensuring that even commercial entities can use OpenMM.

OpenMM needs sustainable federal funding to continue to power the biomolecular modeling and simulation community into the next decade. OpenMM now needs sustainable funding to continue to meet the needs of the biomolecular modeling and simulation community, and grow to deliver the technology needed to drive new innovations in the field as machine learning becomes more tightly integrated at every level. Previously, OpenMM development was funded via NIH support to Vijay Pande, formerly a professor at Stanford University; Pande also founded the Folding@home Distributed Computing project (now governed by the Folding@home Consortium [32], of which co-I Chodera is a member). Specifically, OpenMM development was supported Simbios, funded via NIH Roadmap for Medical Research Collaborative Grant U54 GM072970 (which ended in 2013) and subsequently by NIH grant R01 GM062868 (which ended in 2018 when Vijay Pande transitioned to a General Partner at Andressen Horowitz, stepping down from his position to become an Adjunct Professor at Stanford). This has left OpenMM without long-term financial support; funding for the current lead developer (Peter Eastman) will expire 1 Sep 2020.

Without NIH funding, OpenMM maintenance and development will cease, and the large biomolecular modeling and simulation community that depends on OpenMM will be left without a vital tool for quantitative bioscience. While other MD packages have recently added GPU support, no existing alternative code can fulfill all the roles that OpenMM currently does in enabling research, allowing legacy codes to exploit modern hardware, and enabling the rapid development of new molecular modeling and simulation applications, and development of a new equivalent framework that can fill these roles from scratch would cost millions of dollars.

A new leadership team will transition OpenMM toward a distributed development and community governance model while driving the next generation of technology innovations. While Vijay Pande remains involved in the role of collaborator (see Letter) to ensure that OpenMM can drive new research innovations in the emerging field of machine learning for bimolecular modeling, this proposal is led by a new investigator team that has been deeply engaged with OpenMM throughout its lifetime: **PI Markland** (Stanford) has led new developments in path integral molecular dynamics methods within OpenMM; **co-I Chodera** (MSKCC) has been deeply involved in extending OpenMM to treat small molecules and for use in free energy calculations; and **co-I de Fabritiis** (Barcelona) is a GPU MD pioneer and leader in machine learning for biomolecular modeling. Senior personnel **Peter Eastman** (lead OpenMM developer) will continue to lead development as OpenMM transitions to a distributed development model, bringing in new developers at MSKCC and Barcelona and cultivating community developer contributions. In addition, the leadership team has formed a new **OpenMM Consortium** consisting of leaders in the field that actively develop with or use OpenMM and will help steer future development and community engagement plans to ensure OpenMM can best meet the needs of the community. Together, the investigators and OpenMM Consortium will ensure OpenMM remains a key tool driving new discoveries and innovations within the biomolecular modeling community.

INNOVATION

This proposal aims to go much further than simply *sustain* development of OpenMM with the addition of new features and optimization to take advantage of new graphics processor hardware. Here, **we propose to radically extend OpenMM to exploit and drive new innovations within the machine learning community for biomolecular modeling that promises to revolutionize the practice of biomolecular modeling in a few short years**.

Machine learning is driving a technological revolution in science and engineering. Advances in a key machine learning (ML) technology—deep learning, in which differentiable universal function approximators composed of simple layers can easily be trained using automatic differentiation techniques—has driven a tidal wave of technology development that is transforming many fields of science [33]. To be successful, ML methods require three main ingredients: (1) a **useful representation** of the system to be modeled that allows a useful relationship to be learned from a tractable amount of data; (2) **large datasets** that can be used to train and asses the models, and (3) **fast hardware-accelerated implementations** for both training and using the models in practical applications.

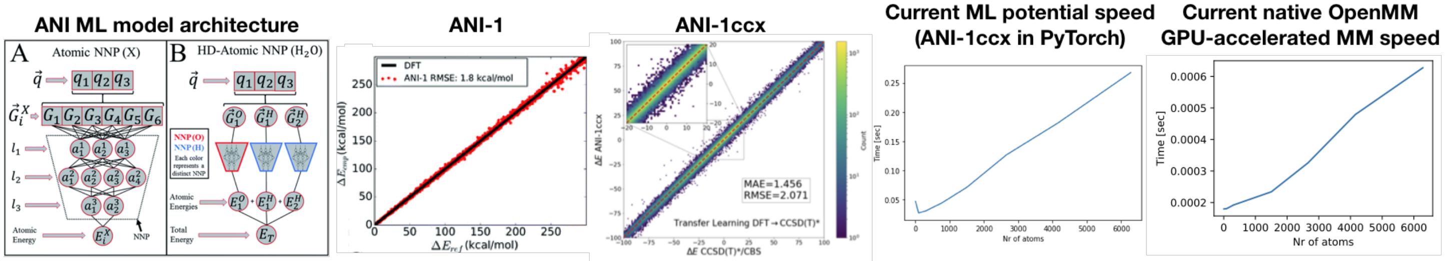

Figure 1. **Quantum machine learning (QML) potential models like ANI-1 can reproduce quantum chemical potential energies with high accuracy, but current implementations are too slow to be practical for biomolecular simulation.** The ANI-1 model [34] (left) achieves excellent accuracy compared to DFT quantum chemical calculations (middle). Second-generation ANI-1ccx [35] achieves high accuracy compared to very high-level CCSD(T) calculations (middle). However, these methods are currently 200× slower than MM potentials in simulating even alanine dipeptide in a water droplet on GPUs such as the NVIDIA GTX 1080 (right). While ML frameworks like PyTorch and TensorFlow make effective use of the GPU for matrix operations (and hardware will keep accelerating these operations), current implementations of ML potentials lack GPU-accelerated kernels for atomic features, which is the key to OpenMM's extremely high performance on complex traditional MM potentials. We will implement support for these common ML potential features as GPU-accelerated kernels within OpenMM to allow these highly accurate ML potentials to be used at MM speeds as a drop-in replacement for MM potentials in all OpenMM-enabled applications, enabling a whole variety of modeling applications to exploit this revolution in accuracy.

Machine learning holds the potential to transform biomolecular modeling and drug discovery. Several key classes of models have emerged: **Quantum machine learning (QML)** potentials aim to replace the expensive, poorly-scaling machinery of quantum chemistry with fast machine learning (ML) models (often based on deep learning) that can approximate the quantum chemical Born-Oppenheimer potential energy surface to high accuracy at greatly reduced cost [36–38]. The ANI class of models [34, 35] (**Figure 1**), for example, can approximate DFT (ANI-1 [34]) or coupled-cluster (ANI-1ccx [35]) calculations with extremely high accuracy at a fraction of the computational cost. A large number of promising models have been developed in the last few years for learning high-dimensional potential energy surfaces (such as Behler-Parrinello [39], ANI-1 [34, 35], AIMNet [40], SchNet [41, 42]; see **Figure 2** for illustrative examples). The flexibility and speed of these QML models make them poised to supplant slow and complex QM/MM methods [43], while their accuracy and ability to train them making them appealing to potentially supplant MM methods in the near future.

Simultaneously, a variety of **translationally- and rotationally-invariant ML architectures** for predicting molecular properties (such as solubilities or binding affinities) from static structures have emerged that promise to make useful predictions from static structures/ These models include Tensor Field Networks [44], Clebsch-Gordan nets [45–47] and graph convolutional or message-passing networks [48] such as PotentialNet [49].

At the same time, the ML revolution has driven the development of deep learning models as flexible function approximators for learning complex **collective variables** that describe biomolecular structural changes for use in enhanced sampling algorithms for studying processes like conformational changes and protein-ligand binding [50]. These approaches include methods such as neural tICA [51, 52], VAMPnets [53], Anncolvar [54], SGOOP [55], and RAVE [56] that encode their collective variables using some differentiable representation of Cartesian coordinates (to allow chain-rule propagation of forces) connected to a differentiable (deep) model.

To impact the biological sciences, these models need to be deeply integrated with biomolecular modeling frameworks. While published accuracies of these ML methods signal their enormous potential, their impact in bioscience has been significantly limited for multiple reasons that we propose to remedy this project:

- **Barriers to use:** The appeal of forcefield-agnostic simulation packages like OpenMM is that multiple models can easily be used via the same interface, greatly increasing the productivity of researchers and developers. Currently, using a new QML model requires installing author source code, developing a new interface from scratch, and building all the associated algorithms for molecular simulation (such as integrators) atop it. We will solve this problem by creating plugins that allow ML models to be easily imported into OpenMM from a variety of common ML frameworks (**Aim 2**) and a common repository of popular QML potentials that can be used from within OpenMM as drop-in replacements for existing MM potentials (**Aim 4**).
- **Speed:** While QML potentials like ANI-1 [34] are significantly faster and scale better than quantum chemical calculations, they are still incredibly slow compared to typical MM force fields—current implementations are 200–15,000× slower than typical MM force fields (such as AMBER ff14SB [57]) in simulating solvated droplet or periodic systems (**Figure 1**), rendering them impractically slow for all but the most trivial of applications. Implementations that narrow the speed gap between MM and QML is essential for these models to supplant or augment MM potentials to deliver improved accuracy and trainable improvements. We will replicate our successful GPU kernel optimization strategies in optimizing the speed bottleneck—computation of atomic feature vectors—to accelerate QML methods (**Aim 2**).
- **Data availability:** The construction of new QML potentials requires large, high-level quantum chemical datasets that require access to major supercomputing resources and quantum chemistry expertise to generate, putting them out of reach of most researchers. We will collaborate with the MolSSI QCArchive [58] and Folding@home [59] to produce large, high-quality quantum chemical datasets for use in developing improved QML potentials for biomolecular modeling (**Aim 4**).
- **Useful abstractions and feature computation kernel libraries:** The development of new QML models also requires researchers re-implement all the featurization schemes necessary prior to the deep learning model(s) used by these methods to fit data. Here, we will build a library of optimized feature computation ops for popular machine learning frameworks (e.g. PyTorch/TensorFlow) that can be also be used by researchers to accelerate research and development in this domain (**Aim 3**).
- **Hybrid physical/machine learning models:** Many current QML models include only short-range interactions [34, 35, 39], making it challenging for these systems to accurately model molecular systems where long-range interactions are important. While a number of interesting directions are being explored in terms of continuous-convolution [41, 42, 60] or message-passing [40], a promising avenue for development is the use of physical potentials to describe these long-range interactions [61]. In parallel, traditional MM force fields are adopting ML models to describe correlated many-body local effects [62]. We will make it easy to develop and deploy hybrid physical/ML models by enabling OpenMM physical energy terms to easily be combined with QML models in OpenMM simulations and within popular ML frameworks (such as TensorFlow and PyTorch) by developing a library of custom ops that can easily be used in both OpenMM and ML frameworks (**Aim 3**).
- **Flexibility:** OpenMM will also be able to spur innovation by allowing researchers to flexibly utilize these capabilities in other places arbitrary functions may be of interest, such as using ML models to define collective variables, biasing potentials, or advanced integrators to explore enhanced sampling schemes (**Aim 2**).

This proposal aims to augment OpenMM with new capabilities that enable it to exploit the machine learning revolution and power the next generation of research and applications in biomolecular modeling and simulation. **Aim 2** describes how QML potentials, ML models for collective variables, and expressive compute graphs for advanced integrators and samplers can be utilized within OpenMM for immediate use by researchers and developers that use OpenMM in their applications, or in legacy codes that use OpenMM to exploit modern hardware. **Aim 3** will enable the use of OpenMM physical forces within popular ML frameworks, as well as provide large quantum chemical datasets that will help accelerate research into QML potentials—especially hybrid potentials. And **Aim 4** will leverage Folding@home to generate new large datasets of high value for building new QML models with sufficiently expanded coverage and improved accuracy to transform biomolecular modeling.

APPROACH

AIM 1: Transition OpenMM from a single-laboratory single-developer-led code to a community governance and distributed developer model to ensure long-term sustainability.

Rationale: OpenMM was developed in a single laboratory (collaborator **Pande**) with support from the NIH, primarily by a single developer (**Eastman**). To ensure OpenMM can sustainably support its extensive user community, we aim to create an OpenMM Consortium that will oversee its future roadmap and governance, as well as recruit and train two additional paid developers within two key OpenMM-focused laboratories (the **Chodera** lab at MSKCC and the **de Fabritiis** lab in Barcelona). In addition, we will seek to recruit more core developers from other laboratories and the biomolecular simulation community at large that depend on OpenMM.

Approach and Milestones: We have formed a new core leadership team—consisting of **PI Markland**, **co-I Chodera**, and **co-I de Fabirtiis**, together with lead OpenMM developer **Eastman** and collaborator Vijay Pande (original OpenMM PI, machine learning pioneer, and Folding@home founder)—which collectively possess extensive expertise in biomolecular simulation, GPU acceleration, and biomolecular machine learning. This leadership team will be responsible for:

- **Recruit new developers to OpenMM:** We will recruit additional developers at MSKCC and Barcelona that will allow core development tasks to be distributed among the development team, ensuring that development is no longer dependent on a single individual. Additionally, the leadership and developer team will seek to recruit additional contributors from other laboratories that depend on OpenMM (such as Andy Simmonett [see Letter] who has made significant contributions already).
- **Develop a public roadmap:** We will develop and maintain a public roadmap for future bugfix, feature, and API-changing OpenMM releases using GitHub Project Boards [63] to ensure the community is well-informed about (and can provide input into) the timing and content of new releases.
- **Accelerate OpenMM on new hardware:** The leadership and developer teams will continue to identify new hardware that would be particularly beneficial for OpenMM to exploit—such as new inexpensive consumer-grade GPUs, new discrete GPUs (such as Intel X^e), integrated hardware accelerators, and tensor processing units (TPUs). Developers—led by Eastman's extensive experience in hardware acceleration—will tailor kernels to continue to exploit the latest hardware (and software frameworks for using them, such as Apple's recent Metal and Intel's oneAPI).
- **Field community-driven feature requests and bug reports:** The development team will clearly and visibly prioritize and address community-provided feature requests and bug reports using GitHub's issue tracker, issue labeling system, and release milestones.
- **Produce enhanced documentation and tutorials:** While OpenMM has extensive user and API documentation and a number of tutorials, we will improve the utility of this documentation and the number of tutorials targeted at new users coming from the machine learning field and users transitioning to OpenMM from other simulation packages could greatly increase the impact and audience of OpenMM. Currently, these resources

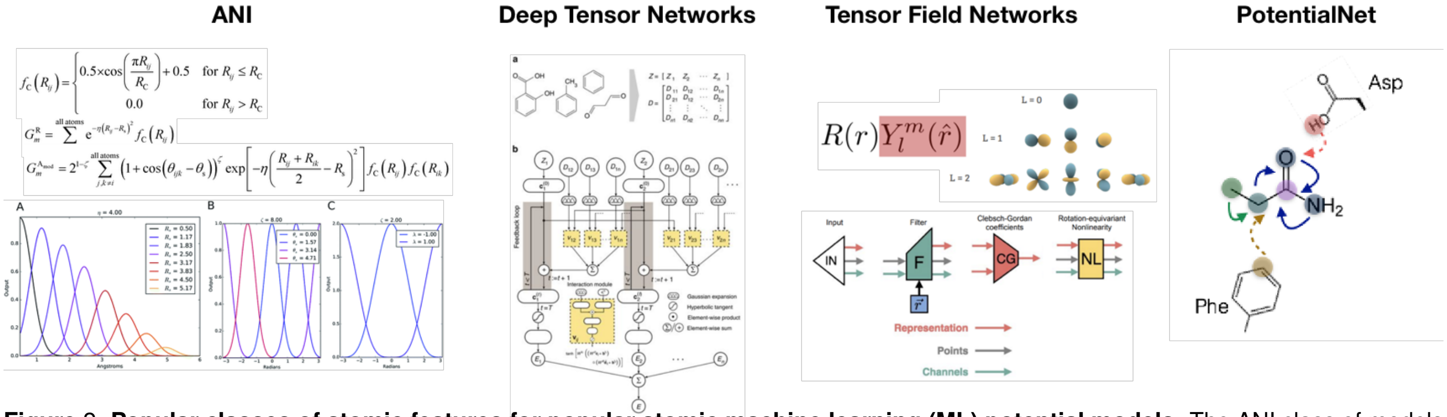

Figure 2. **Popular classes of atomic features for popular atomic machine learning (ML) potential models.** The ANI class of models uses distance- and angle-based features [34, 35]. Continuous convolution networks like Deep Tensor Networks [60] and SchNet [41, 42] use distance-based features. Another class of models, Tensor Field Networks, use spherical harmonics [44–47]. PotentialNet uses a graph convolutional network augmented by distance-dependent edges [49]. To both achieve high performance within OpenMM (**Aim 2**) and enable rapid QML model development in common ML frameworks such as TensorFlow, PyTorch, and JAX (**Aim 3**), we will implement hardware-accelerated compute kernels for these and other common atomic feature computation schemes for use both within OpenMM and as a library of operations ("ops") available within common ML frameworks.

are already highly utilized, with ~1500 unique visitors/month accessing the user guide and API docs and ~500 unique visitors/month accessing the tutorials, suggesting effort spent on improving existing materials will also be highly valuable. We will focus on improving online documentation and materials, which are broadly accessible to the global biomolecular simulation community, and tutorials that can easily be launched in the browser (such as MyBinder and Google Colab) to further minimize barriers to entry.

- **Transition to a community governance model under the OpenMM Consortium:** The leadership team will establish the OpenMM Consortium, consisting of academic researchers, software scientists, and industry scientists with extensive experience with OpenMM and biomolecular modeling to ensure that OpenMM will continue to meet the needs of the biomolecular modeling by setting priorities and integrating best practices appropriately. The leadership team will work closely with the Molecular Sciences Software Institute (MolSSI) [64] (see Letter) to develop a Consortium governance model to make sure that governance follows equitable best practices for a healthy software project that addresses the needs of the biomolecular simulation community.

Preliminary data and feasibility: We have recently consolidated OpenMM into a central GitHub organization independent of any single laboratory [65]. We have also recruited an initial OpenMM Consortium [66], consisting of members of the biomolecular simulation and modeling field who can advise on how OpenMM can best serve the needs of the community (many of which have contributed Letters): Emilio Gallicchio (Brooklyn College), Justin McCallum (University of Calgary), Vijay Pande (Stanford/a16z), Jean-Philip Piquemal (Sorbonne), Jay Ponder (WashU), Julia Rice (IBM Almaden), Josh Rackers (Sandia), Pengyu Ren (UT Austin), Karmen Condic-Jurkic (MSKCC), Andy Simmonett (NIH), and Bill Swope (IBM). The OpenMM Consortium has already held initial virtual meetings to develop a preliminary roadmap for how OpenMM can best serve both the advanced potential function development community (which led to AMOEBA [67] and HIPPO [68]) as well as the quantum machine learning community. Though the current Consortium currently serves in an advisory role, it will ultimately be tasked with developing a governance model as well.

While current benchmarks indicate extremely good performance using CUDA on NVIDIA hardware, with Joint Amber-CHARMM (JAC) DHFR benchmarks with PME of 419 ns/day at 2 fs timesteps and 785 ns/day for 5 fs timesteps with hydrogen mass repartitioning (HMR) [69] on an NVIDIA Titan V [70], new hardware may require the use of new acceleration APIs to fully exploit it. We have therefore refactored the OpenMM CUDA and OpenCL platforms into a single common API to make it easier to rapidly retarget to new acceleration APIs (such as Apple Metal, AMD RocM, and Intel oneAPI) when it becomes clear that this is necessary to achieve state-of-the-art performance on new hardware.

AIM 2: Develop a next-generation hardware-accelerated molecular simulation platform that will enable direct interoperability with machine learning frameworks.

Rationale: Biomolecular modeling is at the cusp of a revolution, with quantum machine learning (QML) potential energy models now able to achieve quantum chemical accuracy using deep learning models that exploit translationally and rotationally invariant features [36–38] (**Figure 1**). While this field is still young, it is unmistakably clear that there will be an explosion in the exploration and application of QML models and hybrid QML/MM models in biomolecular modeling over the next decade, as researchers seek increased accuracy, the ability to tackle problems previously inaccessible or difficult with QM/MM (such as enzyme catalysis and covalent inhibition), and exploit the ability to easily improve these models with quantum chemical and experimental data.

Current implementations are impractically slow—200–15,000× slower than MM simulations within OpenMM—and require writing custom interfaces to glue them to MD codes, which can also result in even more performance loss. To make these methods practically usable by the biomolecular modeling community, we will provide a way to easily import and use ML models developed with popular ML frameworks (TensorFlow [71], PyTorch [72], and JAX [73]) within OpenMM in a flexible manner that allows their use as potentials, collective variables, biasing functions, and integrators. Currently, the major computational bottleneck is that these models must compute fixed-vector-length features from biomolecular systems using ML ops that are not well-suited to dealing with biomolecular data (such as pairlist operations) prior to feeding these feature vectors to highly optimized deep learning models that can exploit hardware that is rapidly improving to accelerate the deep learning stage of the computation (**Figure 3**). We will accelerate these methods by creating custom kernels to eliminate this major computational bottleneck using the same advances that OpenMM has made in accelerating MM potentials.

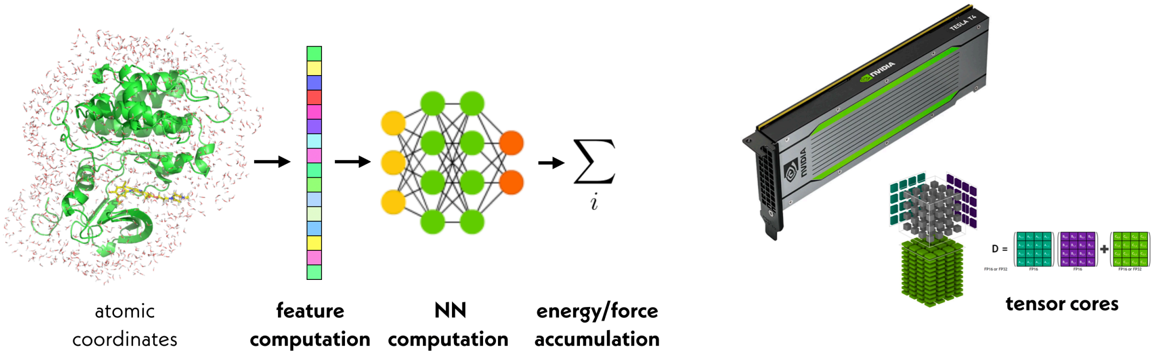

Figure 3. Accelerating atomic feature computation through custom GPU-accelerated ops will eliminate the computational bottleneck hindering widespread adoption of QML potentials. *Left:* The key bottleneck in current QML implementations is the computation of atomic features for biomolecular systems, especially for periodic systems. These operations involve irregular computations like pairlist construction and sums over local atom sets, as well as computation of their derivatives, which are operations that ML frameworks like TensorFlow and PyTorch are not efficient for. By contrast, OpenMM's GPU-accelerated kernels excel at operations like this. We will implement GPU-accelerated custom ops for TensorFlow/PyTorch that speed up these feature computation operations. *Right:* While the subsequent computational stage—neural network computation—may still be computationally costly on current hardware, both hardware and software are aggressively being optimized by hardware vendors and ML package developers to accelerate these computations, including innovations like adding tensor cores. We will therefore primarily focus our effort on accelerating featurization (**Aim 2**) and enabling developers to distill/compress ML models to tune their performance on common accelerator hardware (**Aim 3**).

Approach and Milestones:

- **Develop hardware-accelerated kernels to accelerate QML potentials within OpenMM:** We have developed prototype OpenMM plugins that allow ML models defined within TensorFlow or PyTorch to compute potential energy components or collective variables within OpenMM. These plugins currently use the corresponding ML framework to compute all features directly from Cartesian atomic coordinates, resulting in poor performance compared with OpenMM's GPU-accelerated MM energy functions, which use GPU-accelerated kernels that exploit special properties of atomic coordinates (**Figure 1**). We will develop custom GPU-accelerated operations ("ops") that can be used to replace inefficient atomic feature computation code within these ML frameworks, prioritizing key atomic features used in popular QML models (**Figure 2**): such as Behler-Parrinello [39], ANI-1 [34, 35], AIMNet [40], SchNet [41, 42], Tensor Field Networks [44], Clebsch-Gordan nets [45–47], and graph convolutional or message-passing networks [48] such as PotentialNet [49]. Simultaneously, new generations of hardware will continue to feature optimizations to accelerate the deep learning half of the computation without our help, enabling us to focus on eliminating feature computation as a computational bottleneck. (In **Aim 3**, we discuss how we will also provide model compression tools to help tune models for improved performance.)
- **Enable ML models to be flexibly imported and used in a variety of innovative new ways within OpenMM:** In addition to QML potential support, we will make it easy to import models from ML frameworks such as TensorFlow, PyTorch, or JAX for use wherever a function approximator could be useful, such as collective variables, biasing potentials, or integrators (which are simply methods that update positions and velocities and perform computations on them).

Preliminary data and feasibility: OpenMM has been at the forefront of offering users and developers convenient ways for specifying differentiable functions to flexibly support new potential functions and collective variables while not compromising on speed. The CustomForce facilities of OpenMM—introduced a decade ago [18]—allow users or developers to specify algebraic expressions or spline functions (without the need to write new code) that are symbolically differentiated and compiled into GPU kernels that run at native speed, even allowing differentiation with respect to force field parameters. For example, implementing a Coulomb potential alchemically modified by the parameter `lambda` is as simple as specifying its algebraic form: `'lambda*C*charge1*charge2/r'`
Extending OpenMM to support differentiable machine learning models through compute graphs via TensorFlow, PyTorch, or JAX is conceptually similar, though differentiation is accomplished through forward- or backward-mode automatic differentiation, rather than via symbolic algebra. In both cases, just-in-time compilation provides the ability to rapidly build optimized kernels targeted to multiple hardware accelerators (such as GPUs).

As a proof of concept, we have developed two new plugins that allow models developed in popular machine learning frameworks to be used within OpenMM as quantum machine learning (QML) potentials or collective variables, both of which can be used alongside normal OpenMM physical potential function forces: (1) the **OpenMM TensorFlow**

Plugin [74] allows models exported from TensorFlow [71] to be used within OpenMM, while (2) the **OpenMM PyTorch Plugin** [75] allows models exported from PyTorch [72] to be used within OpenMM.

We have demonstrated that it is possible to import the ANI-1 model [34] defined in TorchANI [76] into OpenMM and simulate a box of water molecules within OpenMM, using our using our prototype PyTorch OpenMM plugin, thereby enabling any applications built on OpenMM to use ANI in place of the wide variety of currently-supported molecular mechanics potentials. While both periodic and non-periodic systems can be simulated, initial performance benchmarks of the unoptimized models show that a box of 215 waters achieves 60 timesteps/second with periodic boundary conditions or 150 timesteps/second without, suggesting the periodic feature computations within the ANI-1 model implemented fully in TorchANI are particularly inefficient. Initial optimization will focus on implementing custom ops within PyTorch that allow the current inefficient periodic and non-periodic feature computation to be accelerated by coalescing multiple inefficient PyTorch operations into highly optimized GPU kernels.

This work will be guided by our machine learning expertise (lead developer **Eastman** and collaborator **Pande** recently co-authored a book on ML for the biosciences [77], and **co-I de Fabritiis** recently achieved highest predictive accuracy in the blinded D3R Grand Challenge 4 [78]), close coordination with the MolSSI Machine Learning team, input from the OpenMM Consortium, and key engagement from laboratories at the forefront of QML (see Letters).

AIM 3: Create a software infrastructure to enable hybrid machine learning / physical model development.

Rationale: Two major software impediments stand in the way of building new QML methods with greatly increased accuracy and expanded coverage for biomolecular systems: (1) Training new ML models on 20M+ molecule datasets [34] is significantly hindered by feature computation—-the slow speed of computing complex input features from atomic coordinates subsequently fed into fast deep learning models (which are already hardware accelerated). (2) While QML models excel at modeling short-range interactions, the need to use physical potential models for long-range interactions (as in PhysNet [61], which explicitly includes long-range electrostatics) is a clear direction for future research. Unfortunately, it is very difficult to bring physical models into modern ML frameworks—especially complex electrostatic models that much be implemented from scratch. By enabling OpenMM's hardware-accelerated feature computation kernels to be used within popular frameworks like TensorFlow and PyTorch, and creating new hardware-accelerated kernels for physical energy terms, we can accelerate research into new ML models not just for fitting quantum chemical energies of biomolecules, but also applications like enzyme design, drug discovery, biotherapeutic optimization, and predicting the functional impact of mutations.

Approach and Milestones:

- **Produce a hardware-accelerated atomic featurization kernel library to accelerate QML development and training:** To accelerate the development, training, and deployment of new QML models, we will package our custom hardware-accelerated kernels for performing common QML atomic featurization operations into a library that can easily be used in popular ML frameworks (PyTorch, TensorFlow, JAX). Besides making it easier for QML developers to easily build models without coding atomic features and accelerating the training process, this will also make it easier for QML developers to deploy their models within OpenMM at full speed, since the same kernels will be used to achieve hardware acceleration within OpenMM.
- **Facilitate the development of hybrid physical/ML models by creating custom ops for physical interaction kernels:** Hybrid physical/ML models such as PhysNet [61] (which predicts the partial charges at every step and uses a classical electrostatics model) are only beginning to be explored because of the difficulties in implementing optimized physical potential energy functions in ML frameworks. To enable rapid exploration of hybrid models, we will also port OpenMM's optimize physical energy function kernels (including advanced physical terms such as those in AMOEBA [67, 79–81], HIPPO [68], CHARMM polarizable force fields [82], and Gaussian electrostatics models [83]) over to optimized ML framework ops to use as building blocks for hybrid models, allowing them to run at high speed both within ML frameworks and OpenMM.
- **Unify ML and MM APIs to accelerate science at the interface:** The difficulty of developing hybrid models is exacerbated by the complexity of methods for setting up models for complex biomolecular systems. We will simplify this process by migrating OpenMM's System description classes to a pure Python object model that will allow users to render a parameterized system either to a native OpenMM Context that can run at full speed on the GPU or to a compute graph representation within ML frameworks like TensorFlow, PyTorch, and JAX that allow the model to be differentiated with respect to arbitrary model parameters for use in force field parameterization.

- **Develop a library of tools for optimizing QML model performance within OpenMM (and other MM packages):** While hardware and software developments outside of this project will continue to deliver increased performance for the neural network stages of QML models (**Figure 3**), we can do more to assist QML model developers in delivering increased performance without compromising accuracy. While a number of techniques based on **model distillation** [84], compression [85], and incremental quantization [86] techniques have been proposed for producing new networks that deliver equivalent accuracy but require greatly reduce FLOP counts, these acceleration techniques have not yet seen use in QML potential refinement and deployment. We will introduce a number of promising approaches into our QML model acceleration and export library to make it easier for QML developers to tune the performance of their models prior to deployment in our QML potential model archive.

Preliminary data and feasibility:

OpenMM's Python API has enabled it to already see significant use within machine learning frameworks. For example, a recent Science paper [87] wrapped OpenMM within a TensorFlow op (without further optimization) to train invertible flows for sampling directly from the Boltzmann distribution. This Aim focuses on finding ways to accelerate this use to make research in this area more practical by eliminating pain points and breaking down the barriers between ML frameworks and physical modeling codes such as OpenMM. We note that others have already demonstrated the feasibility for achieving significant performance optimizations by developing custom hardware-accelerated ops for some physical forces in projects such as jax-md [88] and timemachine [89], but these attempts demonstrate feasibility and do not contain any of the significant infrastructure required by a fully-featured MD code like OpenMM that is aimed at working with biomolecular systems.

AIM 4: Develop community repositories of large quantum chemical datasets and models to accelerate development and deployment of QML potentials.

Rationale: Publicly available quantum chemical datasets of sufficient size for building useful quantum machine learning (QML) potentials covering biomolecules (such as the ANI-1 Dataset [34]) are still highly limited in chemical coverage, conformations, composition, and available levels of quantum theory. To ensure researchers can easily train, validate, and assess new QML potentials—and especially hybrid models that blend ML and physics—we will work in close collaboration with the MolSSI QCArchive project and the Folding@home Consortium to generate large quantum chemical datasets for biomolecules that are useful in both OpenMM and common ML frameworks.

We will additionally make it easy to distribute, modify, and use QML potentials through developing an interchange standard and online repository of QML potentials in close collaboration with the MolSSI QCArchive and Machine Learning team.

Approach and Milestones:

- **Enable facile and reproducible deployment of QML potentials:** OpenMM currently features a flexible ForceField engine using a force field agnostic force field specification, allowing biomolecular systems to quickly and easily be parameterized with a variety of common force fields by just changing the force field specification string (including nearly all force fields commonly distributed with AMBER [29] and CHARMM [21], as well as small molecule force fields such as GAFF [90] and the Open Force Field Initiative SMIRNOFF force fields [91]). We will extend the OpenMM API to similarly make it easy to construct new QML models in a similar fashion, ensuring that a multitude of popular QML potentials can easily be accessed by users by just specifying the released model. We will work closely with the MolSSI QCArchive and Machine Learning teams to establish standards for QML model exchange, as well as a common repository that could be shared by multiple packages, and tools for accessing these models programmatically and via the web (inspired by the QCArchive Machine Learning Datasets Repository [92], ModelZoo [93], and the programmatic flexibility of PyTorch Geometric [94]), and work with developers of these models to encourage them to deposit their models at time of publication. Initially, we will work closely with the TorchANI developers (see Letter) to focus on deploying and accelerating the widely popular ANI family of models [34, 35], which have already seen use in modeling protein-ligand binding for drug discovery through a custom TorchANI-NAMD interface [95].
- **Develop a programmatic interface for exporting/deploying QML models to the QML repository:** To make it easier for QML model developers to deploy their models for use within OpenMM (and other packages that ultimately adopt our interchange format and use the MolSSI QML model repository we co-develop), we will create programmatic interfaces for easily exporting and deploying QML models developed using our custom op library.

- **Develop a Folding@home CPU compute core to generate public quantum chemical datasets to accelerate QML potential development:** We will collaborate with MolSSI software scientists and Folding@home developers (see Letters) to develop a Folding@home CPU compute core to perform quantum chemical calculations to populate the MolSSI QCArchive [58]. The compute core will service quantum chemical (QC) computations through the automated MolSSI QCFractal workflow infrastructure to allow MolSSI scientists to utilize the $\sim$330,000 available CPU cores of the planetary-scale Folding@home distributed computing platform to generate extremely large quantum chemical datasets. We will primarily use the open source quantum chemistry code psi4 [96] via the QCEngine [97] quantum chemistry interface, as our collaborators at MolSSI co-develop these codes. These datasets will be made immediately available to the quantum community for use in constructing QML or hybrid QML/MM potentials (or for other purposes, such as constructing new MM force fields or pure ML models). Because QCArchive contains the memory usage and computation time for all submitted molecules, we will be able to develop a simple predictive model to optimize allocation of work to Folding@home clients capable of efficiently handling its resource requirements.
- **Provide immediate open access to generated QC datasets through the MolSSI QCArchive:** The MolSSI QCArchive [58] and machine leaning (ML) teams are in the process of collecting all significant extant datasets in the the quantum machine learning (QML) potential space. Collectively, these datasets represent $\sim 10^8$–10^9 quantum chemical computations. These quantum chemical datasets are made available to the community programmatically through the QCArchive [58], as bulk downloads via the QCArchive Machine Learning Datasets Repository [92], and on Zenodo for archival purposes. These existing datasets are heterogenous, contain different properties as labels, and have been calculated at various levels of theory. As a first step, the QCArchive project has been recomputing these data at standard levels of theory—semiempricial, Hartree Fock, Density Functional Theory (including LDA, GGA, hybrids, and range-separated hybrids) MP2, and coupled cluster—where feasible. This effort, which currently represents only $\sim 10^7$–10^8 core-hours of computation, would be tremendously bolstered by the resources provided by Folding@Home, which could provide more than 10^9 core-hours/year.
- **Select a variety of chemical datasets relevant to constructing high-accuracy QML models with greatly expanded coverage of chemical space:** MolSSI is already engaging the QML community to allow QCArchive users and QML community partners to generate new QC datasets within QCFractal/QCArchive; this systematic, methodical community engagement program that exploits all the tools at MolSSI's disposal—workshops, software fellows, software scientists, visiting scholars, and their position of leadership in the molecular software sciences—will be key to the long-term success of this effort. Initially, however, we plan to generate key datasets of high priority to kickstart exploration in this area, including the PDB Ligand Expo [98]; Enamine REAL building blocks and molecules containing linkers [99]; small biopolymers representative of peptides, nucleic acids; heterocyclic aromatic scaffolds [100]; small biomolecular dimers; and solvated biomolecules. Pipelines developed by the Open Force Field Initiative and MolSSI for fragmenting, capping, and submitting small molecules to QCFractal will be reused for our purposes [91]. Finally, we will need to extend these quantum chemical datasets to cover more of the periodic table. At the moment, this is due largely to a lack of datasets in the literature beyond the p-block (though we are aware of some transition metal complex datasets); we aim for the community to take a leading role in data selection as the field matures. To fill gaps in user-sourced submissions, generative models (such as g-SchNet [101]) will be applied to existing data in the QCArchive go generate new calculation candidates to keep computation pipelines full.

Preliminary data and feasibility: Co-I **Chodera**, working in close coordination with MolSSI software scientists (see Letter), has conducted a pilot experiment as part of the Open Force Field Initiative [91] to deploy QCFractal for the purposes of generating new quantum chemical data for use in force field parameterization, with the results deposited in QCArchive. Tools for the processing of small molecules, their fragmentation into capped small molecules, generation of input geometries, and processing by QCFractal were developed as part of the Initiative [91]. This pilot experiment took advantage of unused CPU computing cycles at multiple academic institutions, with QCFractal computing on $\sim$300 total cores at any one time. Over the course of several months, this pilot experiment carried out $\sim$8M B3LYP-D3(BJ)/DZVP gradient evaluations with Psi4 [96] on a large variety of small molecules, consuming $\sim$1.5M core-hours, and generating a total of 350K geometry optimizations and 3K torsion drives that were immediately deposited into (and are now freely accessible through) QCArchive [58]. As proposed in this Aim, deploying QCFractal on the planetary-scale Folding@home network will provide access to roughly **three orders of magnitude more compute power** than this pilot experiment, enabling the generation of extremely large ($\sim$100M–10B geometry optimizations) high-value quantum chemical datasets for immediate public use in building machine learning models with greatly expanded chemical coverage and improved accuracy.

The investigators and collaborators also have extensive experience in developing open standards for (bio)molecular models and their exchange. **Eastman** led the development of a force field agnostic specification for OpenMM's force field descriptions that has enabled multiple families of force fields to be easily encoded within a common XML format. Based on this work, Co-I **Chodera** led the development of the SMIRNOFF specification for biomolecular force fields using hierarchical typing and direct chemical perception that allows a wide class of small molecule and biopolymer force fields to be expressed [91, 102]. MolSSI led the development of the QCSchema specification for quantum chemical computations that allows QCEngine [97, 103] to support multiple back-end quantum chemistry codes via a single standard format.

Bibliography and References Cited

[1] Frenkel, D. and Smit, B.: Understanding molecular simulation: from algorithms to applications, volume 1. Elsevier, 2001.

[2] Karplus, M. Molecular dynamics simulations of biomolecules, 2002.

[3] Dror, R. O., Dirks, R. M., Grossman, J., Xu, H., and Shaw, D. E.: Biomolecular simulation: a computational microscope for molecular biology. Annual review of biophysics. 41: 429–452, 2012.

[4] Klepeis, J. L., Lindorff-Larsen, K., Dror, R. O., and Shaw, D. E.: Long-timescale molecular dynamics simulations of protein structure and function. Current opinion in structural biology. 19(2): 120–127, 2009.

[5] Stary, A., Kudrnac, M., Beyl, S., Hohaus, A., Timin, E., Wolschann, P., Guy, H. R., and Hering, S.: Molecular dynamics and mutational analysis of a channelopathy mutation in the iis6 helix of cav1. 2. Channels. 2(3): 216–223, 2008.

[6] Wan, S. and Coveney, P. V.: Molecular dynamics simulation reveals structural and thermodynamic features of kinase activation by cancer mutations within the epidermal growth factor receptor. Journal of computational chemistry. 32(13): 2843–2852, 2011.

[7] Lemkul, J. A. and Bevan, D. R.: The role of molecular simulations in the development of inhibitors of amyloid β-peptide aggregation for the treatment of alzheimer's disease. ACS chemical neuroscience. 3(11): 845–856, 2012.

[8] Bottaro, S. and Lindorff-Larsen, K.: Biophysical experiments and biomolecular simulations: A perfect match? Science. 361(6400): 355–360, 2018.

[9] Wang, L., Wu, Y., Deng, Y., Kim, B., Pierce, L., Krilov, G., Lupyan, D., Robinson, S., Dahlgren, M. K., Greenwood, J., and others, : Accurate and reliable prediction of relative ligand binding potency in prospective drug discovery by way of a modern free-energy calculation protocol and force field. Journal of the American Chemical Society. 137(7): 2695–2703, 2015.

[10] Sherborne, B., Shanmugasundaram, V., Cheng, A. C., Christ, C. D., DesJarlais, R. L., Duca, J. S., Lewis, R. A., Loughney, D. A., Manas, E. S., McGaughey, G. B., and others, : Collaborating to improve the use of free-energy and other quantitative methods in drug discovery. Journal of computer-aided molecular design. 30(12): 1139–1141, 2016.

[11] Abel, R., Wang, L., Harder, E. D., Berne, B., and Friesner, R. A.: Advancing drug discovery through enhanced free energy calculations. Accounts of chemical research. 50(7): 1625–1632, 2017.

[12] Gapsys, V., Michielssens, S., Seeliger, D., and de Groot, B. L.: Accurate and rigorous prediction of the changes in protein free energies in a large-scale mutation scan. Angewandte Chemie International Edition. 55(26): 7364–7368, 2016.

[13] Bochicchio, A., Jordaan, S., Losasso, V., Chetty, S., Perera, R. C., Ippoliti, E., Barth, S., and Carloni, P.: Designing the sniper: improving targeted human cytolytic fusion proteins for anti-cancer therapy via molecular simulation. Biomedicines. 5(1): 9, 2017.

[14] Salomon-Ferrer, R., Götz, A. W., Poole, D., Le Grand, S., and Walker, R. C.: Routine microsecond molecular dynamics simulations with amber on gpus. 2. explicit solvent particle mesh ewald. Journal of chemical theory and computation. 9(9): 3878–3888, 2013.

[15] Friedrichs, M. S., Eastman, P., Vaidyanathan, V., Houston, M., Legrand, S., Beberg, A. L., Ensign, D. L., Bruns, C. M., and Pande, V. S.: Accelerating molecular dynamic simulation on graphics processing units. Journal of computational chemistry. 30(6): 864–872, 2009.

[16] Eastman, P. and Pande, V. S.: Efficient nonbonded interactions for molecular dynamics on a graphics processing unit. Journal of computational chemistry. 31(6): 1268–1272, 2010.

[17] Eastman, P. and Pande, V. S.: Constant constraint matrix approximation: a robust, parallelizable constraint method for molecular simulations. Journal of chemical theory and computation. 6(2): 434–437, 2010.

[18] Eastman, P. and Pande, V.: Openmm: A hardware-independent framework for molecular simulations. Computing in science & engineering. 12(4): 34–39, 2010.

[19] Eastman, P., Friedrichs, M. S., Chodera, J. D., Radmer, R. J., Bruns, C. M., Ku, J. P., Beauchamp, K. A., Lane, T. J., Wang, L.-P., Shukla, D., and others, : Openmm 4: a reusable, extensible, hardware independent library for high performance molecular simulation. Journal of chemical theory and computation. 9(1): 461–469, 2013.

[20] Eastman, P., Swails, J., Chodera, J. D., McGibbon, R. T., Zhao, Y., Beauchamp, K. A., Wang, L.-P., Simmonett, A. C., Harrigan, M. P., Stern, C. D., and others, : Openmm 7: Rapid development of high performance algorithms for molecular dynamics. PLoS computational biology. 13(7): e1005659, 2017.

[21] Brooks, B. R., Brooks III, C. L., Mackerell Jr, A. D., Nilsson, L., Petrella, R. J., Roux, B., Won, Y., Archontis, G., Bartels, C., Boresch, S., and others, : Charmm: the biomolecular simulation program. Journal of

computational chemistry. 30(10): 1545–1614, 2009.

[22] Harger, M., Li, D., Wang, Z., Dalby, K., Lagardère, L., Piquemal, J.-P., Ponder, J., and Ren, P.: Tinker-openmm: Absolute and relative alchemical free energies using amoeba on gpus. Journal of computational chemistry. 38(23): 2047–2055, 2017.

[23] Anaconda downloads badge for the openmm conda package. https://anaconda.org/omnia/openmm/badges.

[24] Folding@home: Client statistics by OS. https://stats.foldingathome.org/os.

[25] Harvey, M. J., Giupponi, G., and Fabritiis, G. D.: Acemd: accelerating biomolecular dynamics in the microsecond time scale. Journal of chemical theory and computation. 5(6): 1632–1639, 2009.

[26] Openmm api documentation. http://docs.openmm.org/latest/api-python/library.html.

[27] Openmmtools documentation. https://openmmtools.readthedocs.io/.

[28] Anaconda downloads badge for the openmmtools conda package. https://anaconda.org/omnia/openmmtools/badges.

[29] Case, D., Ben-Shalom, I., Brozell, S., Cerutti, D., Cheatham III, T., Cruzeiro, V., Darden, T., Duke, R., Ghoreishi, D., Gilson, M., and others, : Amber 2018; 2018. University of California, San Francisco.

[30] Phillips, J. C., Braun, R., Wang, W., Gumbart, J., Tajkhorshid, E., Villa, E., Chipot, C., Skeel, R. D., Kale, L., and Schulten, K.: Scalable molecular dynamics with namd. Journal of computational chemistry. 26(16): 1781–1802, 2005.

[31] Van Der Spoel, D., Lindahl, E., Hess, B., Groenhof, G., Mark, A. E., and Berendsen, H. J.: Gromacs: fast, flexible, and free. Journal of computational chemistry. 26(16): 1701–1718, 2005.

[32] Folding@home Consortium. https://foldingathome.org/about/the-foldinghome-consortium/.

[33] LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning. nature. 521(7553): 436–444, 2015.

[34] Smith, J. S., Isayev, O., and Roitberg, A. E.: Ani-1, a data set of 20 million calculated off-equilibrium conformations for organic molecules. Scientific data. 4: 170193, 2017.

[35] Smith, J. S., Nebgen, B. T., Zubatyuk, R., Lubbers, N., Devereux, C., Barros, K., Tretiak, S., Isayev, O., and Roitberg, A. E.: Approaching coupled cluster accuracy with a general-purpose neural network potential through transfer learning. Nature communications. 10(1): 1–8, 2019.

[36] Behler, J.: Perspective: Machine learning potentials for atomistic simulations. The Journal of chemical physics. 145(17): 170901, 2016.

[37] Von Lilienfeld, O. A.: Quantum machine learning in chemical compound space. Angewandte Chemie International Edition. 57(16): 4164–4169, 2018.

[38] Schmitz, G., Godtliebsen, I. H., and Christiansen, O.: Machine learning for potential energy surfaces: An extensive database and assessment of methods. The Journal of chemical physics. 150(24): 244113, 2019.

[39] Behler, J. and Parrinello, M.: Generalized neural-network representation of high-dimensional potential-energy surfaces. Physical review letters. 98(14): 146401, 2007.

[40] Zubatyuk, R., Smith, J. S., Leszczynski, J., and Isayev, O.: Accurate and transferable multitask prediction of chemical properties with an atoms-in-molecules neural network. Science advances. 5(8): eaav6490, 2019.

[41] Schütt, K., Kindermans, P.-J., Felix, H. E. S., Chmiela, S., Tkatchenko, A., and Müller, K.-R.: Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In Advances in neural information processing systems. 2017, pp 991–1001.

[42] Schütt, K. T., Sauceda, H. E., Kindermans, P.-J., Tkatchenko, A., and Müller, K.-R.: Schnet–a deep learning architecture for molecules and materials. The Journal of Chemical Physics. 148(24): 241722, 2018.

[43] Senn, H. M. and Thiel, W.: Qm/mm methods for biomolecular systems. Angewandte Chemie International Edition. 48(7): 1198–1229, 2009.

[44] Thomas, N., Smidt, T., Kearnes, S., Yang, L., Li, L., Kohlhoff, K., and Riley, P.: Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. arXiv preprint arXiv:1802.08219. 2018.

[45] Kondor, R., Lin, Z., and Trivedi, S.: Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. In Advances in Neural Information Processing Systems. 2018, pp 10117–10126.

[46] Kondor, R.: N-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. arXiv preprint arXiv:1803.01588. 2018.

[47] Anderson, B., Hy, T. S., and Kondor, R.: Cormorant: Covariant molecular neural networks. In Advances in Neural Information Processing Systems. 2019, pp 14510–14519.

[48] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E.: Neural message passing for quantum chemistry. In Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017, pp 1263–1272.

[49] Feinberg, E. N., Sur, D., Wu, Z., Husic, B. E., Mai, H., Li, Y., Sun, S., Yang, J., Ramsundar, B., and Pande,

V. S.: Potentialnet for molecular property prediction. ACS central science. 4(11): 1520–1530, 2018.

[50] Wang, Y., Ribeiro, J. M. L., and Tiwary, P.: Machine learning approaches for analyzing and enhancing molecular dynamics simulations. Current Opinion in Structural Biology. 61: 139–145, 2020.

[51] Wehmeyer, C. and Noé, F.: Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. The Journal of chemical physics. 148(24): 241703, 2018.

[52] Sultan, M. M. and Pande, V. S.: Automated design of collective variables using supervised machine learning. The Journal of chemical physics. 149(9): 094106, 2018.

[53] Mardt, A., Pasquali, L., Wu, H., and Noé, F.: Vampnets for deep learning of molecular kinetics. Nature communications. 9(1): 1–11, 2018.

[54] Trapl, D., Horvaćanin, I., Mareška, V., Özçelik, F., Spiwok, V., and Unal, G.: Anncolvar: approximation of complex collective variables by artificial neural networks for analysis and biasing of molecular simulations. Frontiers in molecular biosciences. 6: 25, 2019.

[55] Pramanik, D., Smith, Z., Kells, A., and Tiwary, P.: Can one trust kinetic and thermodynamic observables from biased metadynamics simulations?: Detailed quantitative benchmarks on millimolar drug fragment dissociation. The Journal of Physical Chemistry B. 123(17): 3672–3678, 2019.

[56] Wang, Y., Ribeiro, J. M. L., and Tiwary, P.: Past–future information bottleneck for sampling molecular reaction coordinate simultaneously with thermodynamics and kinetics. Nature communications. 10(1): 1–8, 2019.

[57] Maier, J. A., Martinez, C., Kasavajhala, K., Wickstrom, L., Hauser, K. E., and Simmerling, C.: ff14sb: improving the accuracy of protein side chain and backbone parameters from ff99sb. Journal of chemical theory and computation. 11(8): 3696–3713, 2015.

[58] The MolSSI Quantum Chemistry Archive: A central source to compile, aggregate, query, and share quantum chemistry data. https://qcarchive.molssi.org/.

[59] Folding@home. https://foldingathome.org/.

[60] Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., and Tkatchenko, A.: Quantum-chemical insights from deep tensor neural networks. Nature communications. 8(1): 1–8, 2017.

[61] Unke, O. T. and Meuwly, M.: Physnet: a neural network for predicting energies, forces, dipole moments, and partial charges. Journal of chemical theory and computation. 15(6): 3678–3693, 2019.

[62] Best, R. B., Mittal, J., Feig, M., and MacKerell Jr, A. D.: Inclusion of many-body effects in the additive charmm protein cmap potential results in enhanced cooperativity of α-helix and β-hairpin formation. Biophysical journal. 103(5): 1045–1051, 2012.

[63] OpenMM GitHub Project Boards, howpublished = https://github.com/openmm/openmm/projects.

[64] The Molecular Sciences Software Institute (MolSSI). http://molssi.org.

[65] OpenMM GitHub Organization. https://github.com/openmm.

[66] The OpenMM Consortium. http://openmm.org/about.html#consortium.

[67] Ponder, J. W., Wu, C., Ren, P., Pande, V. S., Chodera, J. D., Schnieders, M. J., Haque, I., Mobley, D. L., Lambrecht, D. S., DiStasio Jr, R. A., and others, : Current status of the amoeba polarizable force field. The journal of physical chemistry B. 114(8): 2549–2564, 2010.

[68] Rackers, J. A. A Physics-Based Intermolecular Potential for Biomolecular Simulation. PhD thesis, Washington University in St. Louis, 2019.

[69] Hopkins, C. W., Le Grand, S., Walker, R. C., and Roitberg, A. E.: Long-time-step molecular dynamics through hydrogen mass repartitioning. Journal of chemical theory and computation. 11(4): 1864–1874, 2015.

[70] OpenMM 7.4 Benchmarks. http://openmm.org/about.html#benchmarks.

[71] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[72] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., and Garnett, R. (Eds.): Advances in Neural Information Processing Systems 32, pp 8024–8035. Curran Associates, Inc., 2019.

[73] Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., and Wanderman-Milne, S. JAX: composable transformations of Python+NumPy programs, 2018.

[74] The OpenMM TensorFlow plugin. http://github.com/openmm/openmm-nn.

[75] The OpenMM PyTorch plugin. http://github.com/openmm/openmm-torch.

[76] TorchANI. https://github.com/aiqm/torchani.

[77] Ramsundar, B., Eastman, P., Walters, P., and Pande, V.: Deep learning for the life sciences: applying deep learning to genomics, microscopy, drug discovery, and more. " O'Reilly Media, Inc.", 2019.

[78] Parks, C. D., Gaieb, Z., and others, : D3r grand challenge 4: blind prediction of protein-ligand poses, affinity rankings, and relative binding free energies. Journal of computer-aided molecular design. 2020.

[79] Piquemal, J.-P., Perera, L., Cisneros, G. A., Ren, P., Pedersen, L. G., and Darden, T. A.: Towards accurate solvation dynamics of divalent cations in water using the polarizable amoeba force field: From energetics to structure. The Journal of chemical physics. 125(5): 054511, 2006.

[80] Shi, Y., Xia, Z., Zhang, J., Best, R., Wu, C., Ponder, J. W., and Ren, P.: Polarizable atomic multipole-based amoeba force field for proteins. Journal of chemical theory and computation. 9(9): 4046–4063, 2013.

[81] Laury, M. L., Wang, L.-P., Pande, V. S., Head-Gordon, T., and Ponder, J. W.: Revised parameters for the amoeba polarizable atomic multipole water model. The Journal of Physical Chemistry B. 119(29): 9423–9437, 2015.

[82] Baker, C. M., Lopes, P. E., Zhu, X., Roux, B., and MacKerell Jr, A. D.: Accurate calculation of hydration free energies using pair-specific lennard-jones parameters in the charmm drude polarizable force field. Journal of chemical theory and computation. 6(4): 1181–1198, 2010.

[83] Cisneros, G. A.: Application of gaussian electrostatic model (gem) distributed multipoles in the amoeba force field. Journal of chemical theory and computation. 8(12): 5072–5080, 2012.

[84] Hinton, G., Vinyals, O., and Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531. 2015.

[85] Cheng, Y., Wang, D., Zhou, P., and Zhang, T.: A survey of model compression and acceleration for deep neural networks. arXiv preprint arXiv:1710.09282. 2017.

[86] Zhou, A., Yao, A., Guo, Y., Xu, L., and Chen, Y.: Incremental network quantization: Towards lossless cnns with low-precision weights. arXiv preprint arXiv:1702.03044. 2017.

[87] Noé, F., Olsson, S., Köhler, J., and Wu, H.: Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. Science. 365(6457): eaaw1147, 2019.

[88] Schoenholz, S. S. and Cubuk, E. D. Jax m.d.: End-to-end differentiable, hardware accelerated, molecular dynamics in pure python. https://github.com/google/jax-md, https://arxiv.org/abs/1912.04232, 2019.

[89] Timemachine: A simplified and differentiable md engine. https://github.com/proteneer/timemachine.

[90] Wang, J., Wolf, R. M., Caldwell, J. W., Kollman, P. A., and Case, D. A.: Development and testing of a general amber force field. Journal of computational chemistry. 25(9): 1157–1174, 2004.

[91] The Open Force Field Initiative: An open source, open science, and open data approach to better force fields. https://openforcefield.org.

[92] The MolSSI QCArchive Machine Learning Datasets Repository. https://qcarchive.molssi.org/apps/ml_datasets/.

[93] Model zoo: Discover open source deep learning code and pretrained models. https://modelzoo.co/.

[94] Fey, M. and Lenssen, J. E.: Fast graph representation learning with PyTorch Geometric. In ICLR Workshop on Representation Learning on Graphs and Manifolds. 2019.

[95] Lahey, S.-L. J. and Rowley, C. N.: Simulating protein-ligand binding with neural network potentials. Chemical Science. 2020.

[96] Parrish, R. M., Burns, L. A., Smith, D. G., Simmonett, A. C., DePrince III, A. E., Hohenstein, E. G., Bozkaya, U., Sokolov, A. Y., Di Remigio, R., Richard, R. M., and others, : Psi4 1.1: An open-source electronic structure program emphasizing automation, advanced libraries, and interoperability. Journal of chemical theory and computation. 13(7): 3185–3197, 2017.

[97] QCEngine: A quantum chemistry program executor and IO standardizer (QCSchema) for quantum chemistry. https://github.com/MolSSI/QCEngine.

[98] Feng, Z., Chen, L., Maddula, H., Akcan, O., Oughtred, R., Berman, H. M., and Westbrook, J.: Ligand depot: a data warehouse for ligands bound to macromolecules. Bioinformatics. 20(13): 2153–2155, 2004.

[99] Enamine REAL. https://enamine.net/library-synthesis/real-compounds/real-compound-libraries.

[100] Pitt, W. R., Parry, D. M., Perry, B. G., and Groom, C. R.: Heteroaromatic rings of the future. Journal of medicinal chemistry. 52(9): 2952–2963, 2009.

[101] Gebauer, N., Gastegger, M., and Schütt, K. Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules. In Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., and Garnett, R. (Eds.): Advances in Neural Information Processing Systems 32, pp 7564–7576. Curran Associates, Inc.,

2019.

[102] Mobley, D. L., Bannan, C. C., Rizzi, A., Bayly, C. I., Chodera, J. D., Lim, V. T., Lim, N. M., Beauchamp, K. A., Slochower, D. R., Shirts, M. R., and others, : Escaping atom types in force fields using direct chemical perception. Journal of chemical theory and computation. 14(11): 6076–6092, 2018.

[103] Quantum chemistry schema: A JSON schema for quantum chemistry. https://molssi-qc-schema.readthedocs.io/.