Integrating Machine Learning with HPC-driven Simulations for Enhanced Student Learning

Vikram Jadhao, JCS Kadupitiya
Intelligent Systems Engineering
Indiana University
Bloomington, Indiana 47408
{vjadhao,kadu}@iu.edu

Abstract—We explore the idea of integrating machine learning (ML) with high performance computing (HPC)-driven simulations to address challenges in using simulations to teach computational science and engineering courses. We demonstrate that a ML surrogate, designed using artificial neural networks, yields predictions in excellent agreement with explicit simulation, but at far less time and computing costs. We develop a web application on nanoHUB that supports both HPC-driven simulation and the ML surrogate methods to produce simulation outputs. This tool is used for both in-classroom instruction and for solving homework problems associated with two courses covering topics in the broad areas of computational materials science, modeling and simulation, and engineering applications of HPC-enabled simulations. The evaluation of the tool via in-classroom student feedback and surveys shows that the ML-enhanced tool provides a dynamic and responsive simulation environment that enhances student learning. The improvement in the interactivity with the simulation framework in terms of real-time engagement and anytime access enables students to develop intuition for the physical system behavior through rapid visualization of variations in output quantities with changes in inputs.

Index Terms—Machine Learning, HPC-driven Simulations, Computational Science, Scientific Computing

I. INTRODUCTION

The use of computational simulations is ubiquitous in investigating phenomena associated with a wide range of disciplines including materials science and engineering, bioengineering, chemistry, chemical engineering, and physics. Simulations have enabled the understanding of microscopic mechanisms underlying several biological and material phenomena such as ion transport across the cell membrane, flow of polymeric liquids, stabilization of colloidal dispersions, and self-assembly of nanostructures [1]–[5]. Classical molecular dynamics (MD) simulations are an important class of simulation approaches that are generalizable to study a broad range of material and chemical systems [1]. In the MD method, Newton's equations of motion for a system of many particles are solved at each timestep to evolve the particle positions, velocities, and forces forward in time. In several applications where the computational complexity per time step is proportional to the square of the total number of particles (system size), MD simulations incur high computational costs. These high costs are typically mitigated by employing high performance computing (HPC) resources and utilizing parallel computing techniques such as OpenMP and MPI. Using HPC-enabled acceleration techniques can dramatically enhance the performance of MD simulations in many cases.

The parallelized MD simulations enable dynamics of systems with a large number of particles over a wide range of input system parameters. In addition to enabling state-of-the-art research, these simulations can be employed as innovative educational tools for teaching materials covered in computational science and engineering courses. However, despite the employment of the optimal parallelization model suited for the size and complexity of the model system, MD simulations can take a relatively long time to furnish accurate information, varying from minutes to days depending on the model system specifics. Primary factors contributing to this scenario are the time delays resulting from the combination of waiting time in a queue on a computing cluster and the actual runtime for the simulation. Given this prognosis, the use of MD simulations has been generally limited to "outside-classroom" activities such as solving homework problems, where the simulation time requirements are easier to meet.

To use MD simulations during in-classroom sessions, the associated computational tool needs to be sufficiently agile to overcome the following educational challenges:

- Provide simulation-based responses to student questions *in real-time*.
- Make the process of explaining underlying scientific concepts seamless by having *rapid access to accurate trends* in the variation of simulation outputs.
- Do *synchronous* simulation-based analysis of model system behavior in real-time with students.
- Provide a *dynamic environment* for students to perform *in silico* experiments during class to learn the concepts by visualizing the system response under different input conditions.

Addressing these educational challenges can improve

the classroom teaching of computational science and engineering concepts, and enhance student learning. These challenges provide the motivation behind the work presented in this paper.

We recently introduced the idea of integrating machine learning (ML) methods with MD simulations to develop "ML surrogates" for MD simulations [6], [7]. We demonstrated that an artificial neural network (ANN) based regression model, trained on data produced by completed runs of a given HPC-accelerated MD simulation, can successfully imitate part or all of that MD simulation. We also showed that performance improvements of several orders of magnitude could be achieved by replacing large-scale HPC simulations with ML surrogates. The central idea and approach were illustrated using MD simulations of ions in nanoconfinement [4]. The ML surrogate was found to accurately predict the ionic distributions and produce the outputs with an inference time of over a factor $10,000$ smaller than the corresponding MD simulation runtime [7], [8].

Our earlier papers focused on the technical details of designing ML surrogates for MD simulations and evaluating the performance metrics associated with the proposed data-driven approach [7], [8]. In this paper, we explore the potential of employing these ML surrogates to address the aforementioned educational challenges and enhance the usability of simulation tools in education. We develop a nanoHUB web application with a GUI that supports both ML surrogate and MD simulation methods to produce simulation outputs. The nanoHUB tool is used for both in-classroom teaching and for solving homework problems associated with two courses offered at Indiana University (IU). The ML surrogate built into the tool is employed extensively during the in-classroom instruction to teach concepts such as self-assembly, ionic behavior near interfaces, nanoscale material design, modeling and simulation, and neural networks. The impact of the use of ML-enhanced tool in student learning is assessed and evaluated by conducting a survey following other assessment studies [9], [10]. Based on the educational evaluation of the tool, we find that the improvement in the interactivity with the simulation framework in terms of dynamic, real-time engagement and anytime access enables enhanced student learning of computational science concepts.

II. RELATED WORK

A. ML for enhancing simulations of material systems

Computational science and engineering is being transformed by the use of ML. In the area of simulations of materials, ML techniques and in particular deep neural networks based methods have been used to predict parameters, generate configurations, classify material properties, and design force fields [11]–[17].

More recently, ML has been used to design surrogate models that can predict specific outcomes of simulations by bypassing part or all of the simulation. For example, the dissociation timescale of compounds was predicted using an ML surrogate for *ab initio* MD simulations by bypassing the time evolution of the particle trajectories [18]. Deep neural networks trained on HPC-generated simulation data were used as an efficient surrogate for molecular simulation to predict adsorption equilibria as a function of thermodynamic state variables [19]. Convolutional neural network based ML "emulators" have been developed to predict simulation outputs such as power spectrum in biogeochemistry [20]. We have also developed surrogates that can predict the outputs (ionic density profiles) of MD simulations of ions in confinement [6], [7], or in another example, compute forces at each timestep in an MD simulation to bypass the expensive force calculation step [21]. While these ML surrogates have enabled remarkable performance improvements to facilitate research investigations, their use for education applications has been relatively unexplored despite their ability to produce outputs in real-time and for a continuous range of input parameters. In this paper, we probe the potential of using ML surrogates for MD simulations to enhance student learning of topics in the area of computational science and engineering.

B. Simulation caching

Generally, the approach to provide simulation output in real-time is to store the previous simulation results in a cache (simulation caching). For example, nanoHUB provides caching as a feature in computational tools created using their Rappture GUI [22]. Cached simulations provide a static environment with pre-selected parameters defining simulations that can be "looked up". This simulation environment offers limited exploration space, interactivity, and responsiveness to the student. To encourage and empower students to directly experiment and explore the model system and associated phenomena, a new approach is needed that delivers an interactive, dynamic, and responsive simulation environment open for wide exploration. We show that ML surrogates are excellent candidates to fulfill this need.

III. BACKGROUND

A. Use of HPC-enabled simulations in education

One of us teaches two courses at IU that have been taken by undergraduate and graduate students with interests in diverse focus areas including nanoscale engineering, bioengineering, computer engineering, chemistry, and physics. The courses feature application-based learning of basic scientific computing concepts and simulation techniques, including the use of parallel computing methods. Applications are designed employing

the state-of-the-art research in nanomaterials engineering covering several material systems such as virus-like particles [3], shape-changing nanocontainers [23], [24], ion channels [4], and polymeric fluids [5]. HPC-enabled MD simulations are key parts of these courses. These simulations serve as important tools for understanding diverse self-assembly phenomena in nanoscale materials [2], predicting material behavior in practical applications [5], and isolating interesting regions of parameter space for experimental exploration [23].

B. GUI-wrapped simulations on nanoHUB

To facilitate the use of simulations by students in classrooms and for solving homework problems, the simulations are deployed as computational tools on nanoHUB [22]. nanoHUB provides free online access and a Jupyter-notebook based GUI wrapper for executing simulation codes. Depending on the input selected by the users on the GUI, the tools are auto-configured to launch simulations on virtual machines or on a supercomputing cluster (managed by Purdue University) such that the associated simulation wait and run time is minimized. The use of the tools are free of cost upon creation of a nanoHUB account, however, users can run only up to 3 concurrent tool sessions. The authors and other research group members have published 6 nanoHUB tools that enable exploration of diverse self-assembly phenomena in nanomaterials: Ions in nanoconfinement [25], Nanosphere electrostatics lab [26], Nanoparticle assembly lab [27], Nanoparticle shape lab [28], Polyvalent nanoparticle binding simulator [29], and Souffle: Virus capsid assembly lab [30]. These tools provide an interactive GUI to students for examining the links between nanomaterial system parameters and their structural and dynamical behavior via renderings of simulation output on the tool canvas. The tools also enable students to learn the workflows associated with a large scientific simulation software ecosystem. Three of the six tools (Ions in nanoconfinement, Nanoparticle assembly lab, and Nanoparticle shape lab) have been employed in teaching materials associated with the aforementioned courses. Some in-classroom lecture videos are available on nanoHUB as educational resources [31], [32]. We have plans to utilize other tools in teaching in future versions of the aforementioned two courses.

C. nanoHUB tool: Ions in nanoconfinement

In general, one can design and integrate ML surrogates with any of the 6 nanoHUB tools shown above. So far, we have designed and integrated an ML surrogate with one of these tools (Ions in nanoconfinement). In this subsection, we briefly describe this tool to help with the analysis of the technical and educational evaluation results discussed in Section V. The nanoHUB tool "Ions in nanoconfinement" [25] enables users to simulate the self-assembly of ions in nanoconfinement created by material surfaces represented as identical planar interfaces. A primitive model of electrolytes is used to model the ions [4], [33]. Simulations are performed using standard MD methods [4], [34]. The inputs associated with the tool include ion valency, ion size, electrolyte concentration, and interface separation. The simulation outputs include the density profiles of ions in confinement. This tool has been employed every semester since Spring 2018 in illustrating concepts in a graduate course (Simulating Nanoscale Systems) and an undergraduate course (Introduction to Modeling and Simulation) at IU. In less than 3 years of its deployment, the tool has been used by over 130 users and run over 3400 times [25].

The tool comes with a hybrid OpenMP/MPI acceleration that enables simulations to be completed within 30 minutes for any combination of input parameters (assuming no waiting time on HPC cluster). In classroom usage, we observed that the fastest simulations took about 10 minutes to provide the converged ionic densities while the slowest ones (generally associated with a larger number of simulation steps and system sizes) took as long as 1 hour. Primary factors contributing to this scenario were the time delays resulting from the combination of waiting time in a queue on a computing cluster and the actual runtime of the MD simulation. Not having rapid access to expected trends in the variation of ionic densities with input parameters made the process of explaining concepts and mechanisms unwieldy and time-consuming. As we demonstrate below, integrating this tool with a ML surrogate improved its overall usability as an educational tool.

IV. ML Surrogates for Simulations

We now describe a general approach, introduced in Refs. [6], [7], that utilizes ML to enable real-time and anytime engagement with simulations, significantly enhancing the potential for their use in both research and education. In this approach, ML surrogate model is designed using data produced by completed runs of a given HPC-accelerated simulation, and then deployed to approximate the complex relationships between the physical input parameters and the output results of simulations. The ML surrogate bypasses the explicit computational evolution of the simulated model components, yielding accurate outputs in much less time and computing costs. Figure 1 shows the overview of this framework. First, the attributes of the model system are fed to the framework. These inputs are used to launch the simulation on the HPC cluster. Simultaneously, these inputs are fed to the ML-based prediction module. Both the simulation and ML methods are designed to extract (infer) the desired output quantities. Error handler
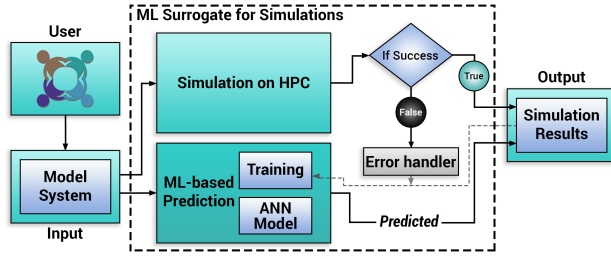
Fig. 1. System overview of the ML surrogate for simulation approach for generating rapid and accurate predictions of simulation outputs for use in classroom teaching.

aborts the simulation program and displays appropriate error messages when a simulation fails due to any pre-defined criteria. At the end of the simulation run, the output quantities are saved for future retraining of the ML model, which occurs after a set number of new successful simulation runs.

In previous papers, we applied this framework to the case of MD simulations of ions in nanoconfinement. The surrogate was trained to learn the relationship between the output distribution of positive ions and 5 input parameters characterizing the ionic system: confinement length h, salt concentration c, positive ion valency z_p, negative ion valency z_n, and ion diameter d. The range of each input parameter were as follows: $h \in (3.0, 4.0)$ nm, $c \in (0.3, 0.9)$ M, $z_p \in 1, 2, 3$, $z_n \in -1$, and $d \in (0.5, 0.75)$ nm. The output quantity was selected to be the distribution of positive ions confined by two identical planar interfaces at $z = -h/2$ and $z = h/2$. For simplicity, using the symmetry of the ionic density around the confinement center $z = 0$, ML surrogate was trained to make predictions characterizing the density of ions in the left half of the confinement (i.e., for $z \in (-h/2, 0)$). The predictions were made at approximately 150 positions; the associated $P \approx 150$ density values were selected as the output parameters (features).

The dataset for training the ANN-based ML surrogate was generated by sweeping over a few discrete values for each of the input and output parameters to create and run 6,864 MD simulations utilizing HPC resources. On average, each MD simulation was run for over 5 million computational steps and took 4200 CPU hours (≈ 36 minutes per simulation). The training dataset creation took approximately 25 days, including the queue wait times on the IU BigRed2 supercomputing cluster. The entire data set was separated into training and testing sets using a ratio of 0.8:0.2. The ANN architecture with 2 hidden layers was implemented in Python using scikit-learn, Keras, and TensorFlow ML libraries [35]–[37] for regression and prediction of $P \approx 150$ continuous

(output) variables. The details of the data generation, preprocessing, ANN feature extraction, and regression are provided in our earlier papers [7], [8].

The ANN-based surrogate model produced the ionic distribution in excellent agreement with explicit MD simulation results [7]. In addition to the high accuracy of ML inferences, the surrogate yielded output results over 10,000 times faster than the parallel MD simulation. The typical ML inference time associated with a prediction of the density profile was ≈ 0.2 seconds (or, almost instantaneous). In strike contrast, the average runtime of the parallel MD simulation to produce a similarly converged output was ≈ 36 minutes.

The overall success rates and rapid inference times associated with predictions made by ML surrogates enable a dynamic and responsive simulation environment for exploration by students in classroom settings. The following capabilities associated with these surrogates are of particular significance in education use:

- Enabling students to verify their learning of features associated with the simulation outputs
- Generating accurate predictions in real-time for unsimulated state points
- Providing a dynamic environment for students to rapidly explore the input-output relationships
- Enabling anytime and anywhere access to simulation results

In the next section, we describe the results associated with the use of simulation tools integrated with ML surrogates in teaching materials associated with computational science and engineering courses.

V. RESULTS

A. Technical evaluation

We first discuss the technical results showing the comparison between the predictions made by the ML surrogate and the outputs of MD simulations. Figure 2 (a) - (d) shows the ionic density profiles predicted by the ML surrogate for a set of 4 systems randomly selected from the entire testing dataset. These systems are: system I $(3, 1, -1, 0.45, 0.7)$, system II $(3.6, 1, -1, 0.9, 0.714)$, system III $(3.7, 3, -1, 0.45, 0.5)$, and system IV $(3.3, 2, -1, 0.3, 0.553)$, where the parentheses list the 5 aforementioned input parameters characterizing the ionic system: confinement length h, positive ion valency z_p, negative ion valency z_n, salt concentration c, and ion diameter d. As the figure indicates, for each system, the ML-predicted density profile is in excellent agreement with the result extracted using MD simulation (ground truth). In addition to the high accuracy, we note that the ML inferences are made in a much shorter time of ≈ 0.2 seconds compared to MD simulations (≈ 36 minutes on average).
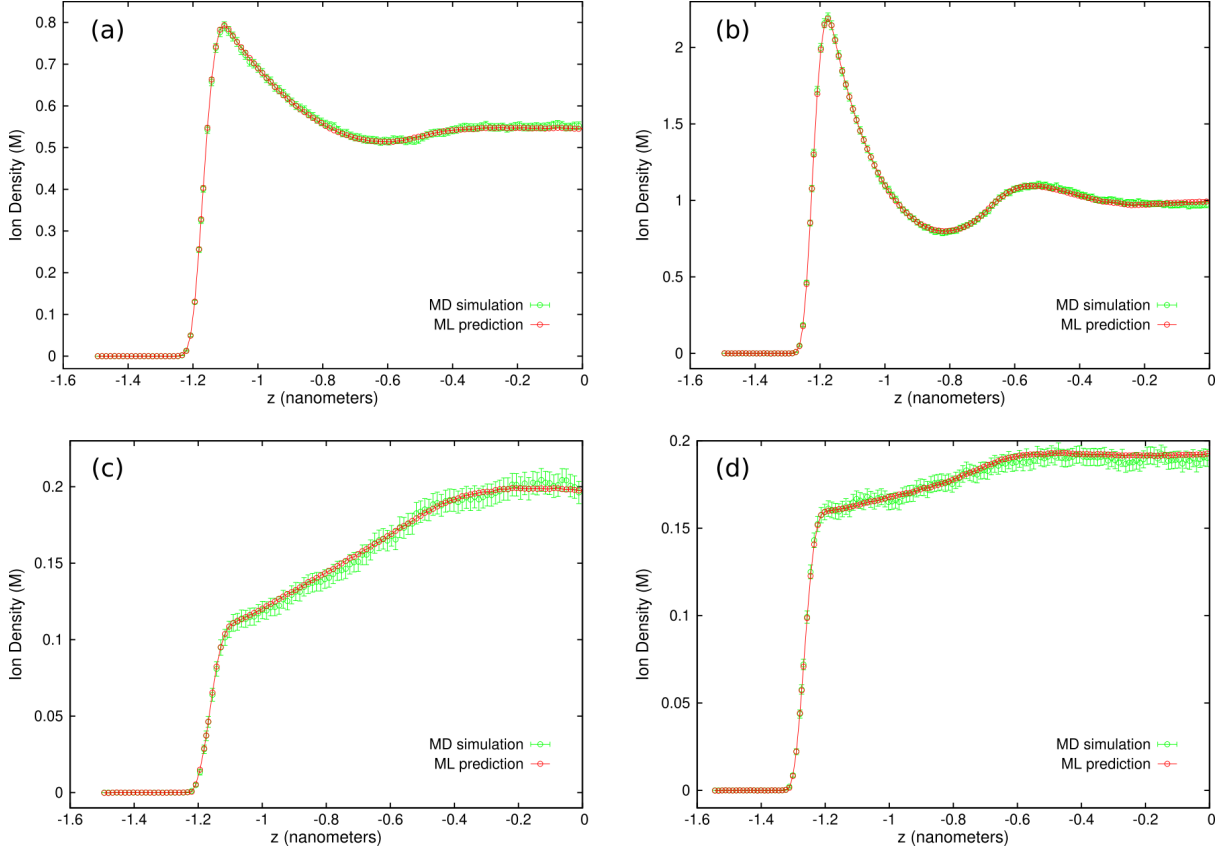
Fig. 2. Ionic density profiles for systems I (a), II (b), III (c), and IV (d) predicted by the ML surrogate (red circles) and extracted with MD simulation (green circles with errorbars). See main text for system definitions. For each system, the ML-predicted density profile is in excellent agreement with the simulation result.

Motivated by the good agreement between ionic densities generated via ML surrogate and MD simulations as well as the remarkable performance enhancement resulting from the use of ML surrogates, we integrated the ML surrogate with the nanoHUB tool "Ions in nanoconfinement". The ML-enhanced tool was deployed on nanoHUB in October 2019. Figure 3 shows the Jupyter python notebook based GUI of the deployed tool. Users are provided with the choice to click "Run" and "Predict using ML" buttons simultaneously or separately depending on the desired information. "Predict using ML" activates the ML surrogate which predicts half of the density profile instantaneously; the result is available in the "Prediction Graph" tab as well as in the "Positive Ion Density" tab. Users can enable ML surrogate any time by clicking the "Predict using ML" button to access the ML-predicted ionic density profile. Clicking the "Run" button instructs the execution engine to either submit a job on an HPC cluster (if the "HPC mode" button is checked) or run the simulation on a VM. When the simulation is over, the execution engine passes the generated data to be plotted on the

"Positive Ion Density" and "Negative Ion Density" tabs. For illustration purposes, Figure 3 also shows the final density plot obtained using the integrated MD and ML method for the input parameters $h = 3.0$ nm, $z_p = 1$, $z_n = -1$, $c = 0.5$ M, and $d = 0.714$ nm. The ML prediction is shown as an overlay in the "Positive Ion Density" tab along with the result of the MD simulation.

B. Educational evaluation

An accurate and rapid assessment of ionic distributions in confinement by the ML surrogate enables in-classroom instruction of several important concepts such as interfacial effects, self-assembly in nanoscale systems, and the intimate connection between solution conditions and the material assembly behavior. For example, by using the ML surrogate, students can instantaneously record changes in the ionic structure as the salt concentration c is tuned. Figure 4 shows a selected subset of ionic density profiles predicted by the ML surrogate for different $c = 0.3, 0.5, 0.7, 0.9$ M. Other input parameters are fixed to $h = 3.0$ nm, $z_p = 1$, $z_n = -1$, and $d = 0.5$ nm. By performing *in silico* experiments in rapid succession using the ML surrogate, students can
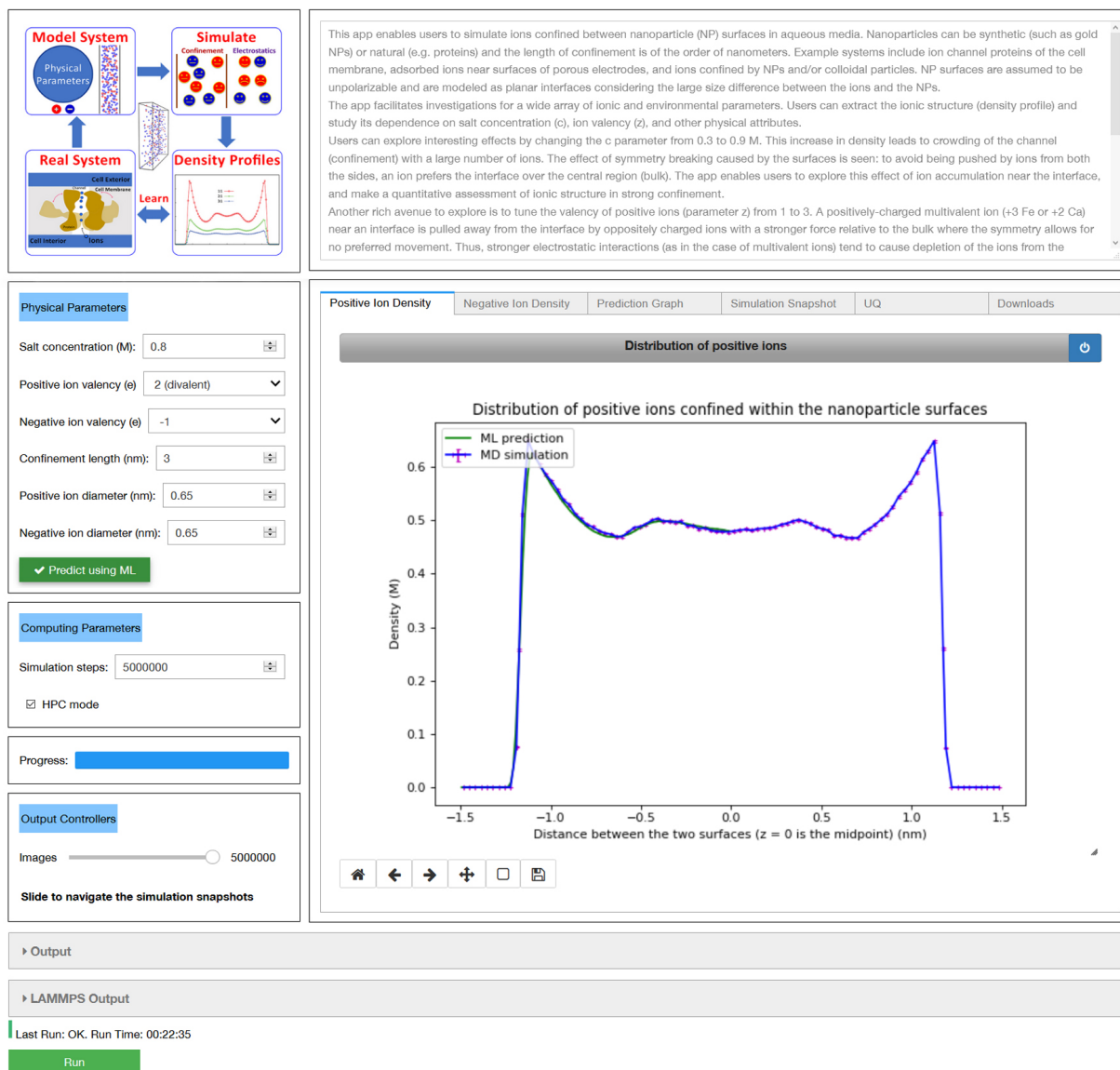
Fig. 3. GUI of the ML-enhanced "Ions in nanoconfinement" nanoHUB tool [25]. The GUI shows the density profile predicted by the ML surrogate for half of the position values (green line) and the result extracted via MD simulation (red markers) for an example ionic system.

readily visualize the response of the ionic system under changes in c. For example, students learn that increasing salt concentration leads to the accumulation of ions near the interface (higher peaks in the ionic density) or to the emergence of more modulations in the density profile. Both these observations inferred by the ML surrogate follow the expected behavior in these systems as reported and elucidated in previous work [4].

As noted before, one of the authors regularly teaches two courses at IU: 1) Simulating nanoscale systems (Fall semester; course for graduate students and advanced undergraduate students) and 2) Introduction to modeling and simulation (Spring semester; undergraduate course).

The students in these courses learn computational model development, simulation techniques such as molecular dynamics, data analysis and visualization, computational materials science concepts such as self-assembly and interfacial phenomena, parallel computing methods, and engineering applications of simulations. The learning is facilitated by having students perform HPC-based simulations that enable the extraction of structure-property relationships in materials at the nanoscale. Students also become familiar with important practical aspects of research in scientific computing such as scalability, time discretization, convergence, model resolution, and simulation accuracy. The key learning objectives of these
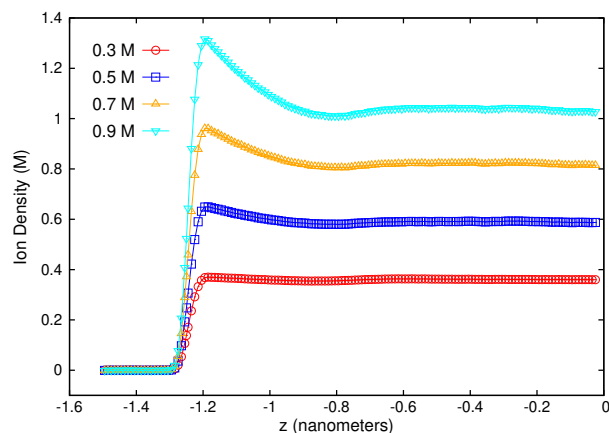
Fig. 4. Density profiles of confined positive ions for different salt concentration $c = 0.3, 0.5, 0.7, 0.9$ M predicted by the ML surrogate.

courses are:

1) Students will understand and develop computational models of real materials.
2) Students will understand materials science concepts and the effects of altering environmental conditions on structure-property relationships.
3) Students will develop simulation methods and solve engineering problems by analyzing computational data.
4) Students will learn the use of parallel computing and data-driven methods in computational science.
5) Students will learn the use of web-based computational tools and understand the scientific software ecosystem.

nanoHUB computational tools are key parts of these courses as they help facilitate the use of simulations by students via a user-friendly, web application requiring no software installation to run the simulations. The ML surrogate was integrated into the nanoHUB tool "Ions in nanoconfinement" and the enhanced tool was pilot tested in Course 1 in Fall 2019. Six students majoring in different fields, including nanoengineering, computer engineering, and chemistry took the course. The tool was also used in Course 2 by 15 students in Spring 2020. Students used the tool during in-classroom lectures as well as to solve homework problems. The tool was actively employed by the instructor in the classroom to help students develop an intuitive understanding of the ionic system behavior via rapid experimentation and visualization of changes in ionic structure.

Below we enumerate a subset of the learning outcomes of these courses in order to provide the context for the results of the tool evaluation discussed in the remainder of this section. When students complete the aforementioned two courses they should be able to:

1) Develop scale-appropriate and computationally-efficient models of real / experimental systems.
2) Develop an in-depth understanding of computational materials science concepts such as self-assembly and structure-property relationships.
3) Develop simulation methods and apply them to solve engineering problems.
4) Use parallel computing methods to enhance computational simulations of nanoscale materials.
5) Use web-based computational tools and understand the associated scientific workflow.

To assess the impact of the use of the ML-enhanced tool and associated simulations on student learning, a tool evaluation survey was conducted at the end of the Fall 2019 semester, where six students provided feedback on their experience with the tool. The survey questions were constructed following similar educational evaluation studies [9], [10], [38]–[43], and comprised of both rating-based and text-response-based questions.

First, a set of 10 questions tabulated in Table I asked the students to rate the simulation tool in terms of different features such as user-friendliness, clarity, utility, consistency etc. Participant ratings as a percentage for the 10 questions listed in Table I are shown in the form of a bar graph in Figure 5. The rating scale was from 1 to 5, where higher scores represent higher ratings. We received a total of 60 rating responses for these 10 questions (every question was answered by all the 6 students). Based on the responses received for all 10 questions, the mean response rating was 4.26 with a variance of 0.39, indicating that on average the students evaluated the simulation tool close to the highest rating of 5.0. More specifically, in terms of user-friendliness, convenience, GUI layout, and consistency, students rated the tool at 4 or higher. We also asked the students how many times they used the nanoHUB tool in the classroom during the lecture sessions. 16.7% of the students responded that they have used it more than 20 times, while 50% said they used the tool between 10 and 20 times (Figure 6). This variation in the tool usage indicates that the ML surrogate was used for diverse purposes ranging from verifying the expected results of the simulation to recording the evolution of the output quantities by tuning the input parameters.

Next, we asked a series of text-response-based questions. We discuss a few of these questions below. Students were asked what aspects of the online simulation tool were useful and valuable to them. The students highlighted that the simulation tool helped them increase the conceptual and practical understanding of the nanoscale simulations due to the user-friendly interface, ML-enabled instantaneous predictions, and availability of multidimensional input choices. For example, here is an excerpt from a student response: "*ML provided the quick answer when that was needed. Easier to use for*

ID	Question
Q1	Were the use of simulations in the class valuable in learning concepts?
Q2	Were the learning objectives regarding the use of the nanoHUB tool clear?
Q3	Rate the tool in terms of user-friendliness.
Q4	Rate the tool in terms of convenience (in terms of how fast the results were inferred by ML).
Q5	Rate the tool in terms of accuracy (as compared with MD results).
Q6	Rate the tool in terms of use for in-class conceptual understanding.
Q7	Rate the tool in terms of use for homework problem solving.
Q8	Rate the tool in terms of GUI layout.
Q9	Rate the tool in terms of quality.
Q10	Rate the tool in terms of consistency.

Fig. 6. A pie chart showing how many times students used the ML-enhanced simulation tool in the classroom. Nearly 16% of the students used the tool over 20 times.
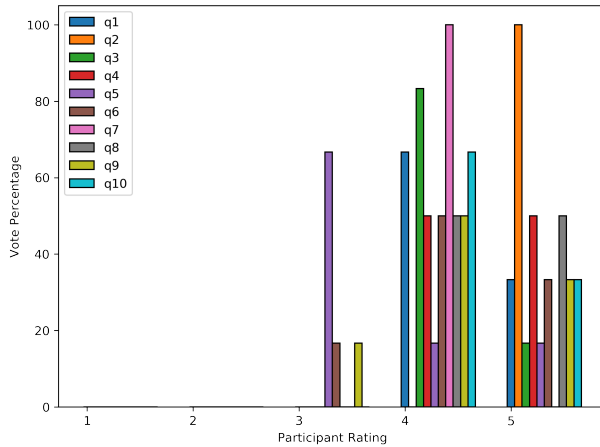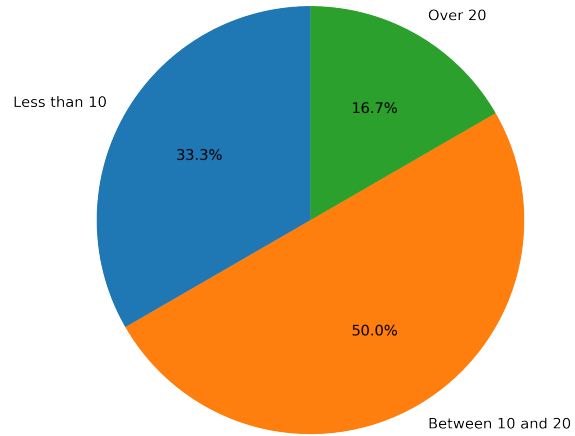

Fig. 5. Participant ratings (as a percentage) for the questions (Table I) used in the survey for evaluating the ML-enhanced tool.

single simulation than accessing supercomputer.*"*

Students were asked to compare simulation-driven classroom teaching experience with a non-simulation-driven classroom teaching experience. 83% of the students enjoyed simulation-driven teaching of computational science concepts stating that "*simulations aid to understanding the concept taught in the class more clearly*" and "*it allows students to be the researchers and experimenters in the sense of using these tools to generate our results for our assignments*". The rest (17%) still preferred the simulation-driven classroom teaching experience but stated that they felt there were occasions with extra downtime in the classroom because of waiting for cluster resources to run the simulations, and they suggested that "*the cluster waiting time needs to be filled with useful content*".

Students were also asked to isolate what aspects of the nanoHUB online tool were most useful to them. 80% of the responses indicated that the students like the ML prediction feature. Here is an excerpt from a student response: "*the predicted machine learning aspect was beneficial because it was very accurate with the simulated results, so if need be one do not have to wait for the simulation to finish computing to know what the results would have been*". The survey responses also indicated that the students enjoy the freedom to probe the system behavior by tuning several model parameters, and they find the output graphs helpful.

Finally, students were asked to provide suggestions to improve the ML-enhanced simulation tool. 66.6% of the students provided feedback to improve the tool, while 33.4% said that they do not have any suggestions to improve the tool. Some suggestions were: "*allowing users to download the ML prediction graph*" and "*providing 3D snapshots of the simulation*". The tool has been updated based on these useful suggestions and the latest version provides options to download the ML prediction result and visualize the snapshots of ions in confinement. Some suggestions such as "*graph updates do not always happen when changing values and toggling ML, especially after full simulation was run*" have not yet been implemented. These are related to the GUI rendering issues which we plan to resolve in the future working with the nanoHUB team.

The ML surrogate helped the instructor of these courses in pursuing many of the aforementioned learning objectives by resolving the 4 key educational challenges outlined in Section I. The remarkable agility of the

surrogate in yielding the predictions enabled the instructor to respond to student questions in real-time via live demonstration using the ML surrogate. The instructor was also able to perform the analysis of the model system behavior synchronously with students. By performing *in silico* experiments in rapid succession using the ML surrogate and visualizing the results on the tool canvas in real-time helped the instructor to illustrate mechanisms underlying the response of the model system to changes in control parameters. Having rapid access to accurate trends in the variation of simulation output with input parameters significantly eased the process of explaining difficult materials science concepts such as self-assembly. Further, the use of the ML-enhanced tool helped in illustrating several practical aspects of scientific computing including the tradeoffs between simulation accuracy, scalability, and efficiency.

VI. Discussion and Conclusion

In this paper, we explored the potential of using ML surrogates for HPC-enabled simulations to address several educational challenges in teaching computational science and engineering courses. The ML surrogate yields predictions in excellent agreement with simulation, but at far less time and computing costs, delivering a dynamic and responsive simulation environment for rapid exploration by students in classroom settings. We developed a web application on nanoHUB that supported both HPC-enabled simulation and the ML surrogate methods to produce simulation outputs.

The nanoHUB tool was used for both in-classroom instruction and for solving homework problems associated with two courses covering topics on computational materials science, modeling and simulation, and engineering applications of HPC-enabled simulations. The educational utility of the tool was evaluated using a survey that was answered enthusiastically by the students. Overall, the learning outcomes, survey results, and the feedback from students in the classroom indicate that the ML surrogate helped resolve key educational challenges and enabled the realization of many learning objectives of the courses. Survey responses showed that the ML-enhanced tool is well-accepted among students and scored very high marks on convenience, user-friendliness, and consistency. Students also provided constructive feedback to improve the tool further in order to ensure its future success. The improvement in the interactivity with the simulation framework in terms of real-time engagement and anytime access enhanced the student learning of computational science concepts.

Results from this investigation are encouraging and we expect the ML surrogate approach to be broadly applicable. We plan to explore the development of ML surrogates to predict outputs of other simulations including those of shape-changing nanoparticles [23], [28], [32] and different types of Monte Carlo simulations [1], [44], [45]. Another line of future work is to explore ways to reduce the training costs of the ML surrogates and probe their potential in predicting outputs outside the pre-defined range of training datasets. We realize that our initial survey involved a small number of students; we plan to conduct more surveys in the future and continue the evaluation of the educational utility of the tools.

We note that the integration of the ML surrogate in a computational tool hosted on nanoHUB exposes this approach to a much broader community of students, educators, and researchers. nanoHUB is the largest online resource for educational materials in nanotechnology [22], hosting over 500 web applications for launching simulations and serving over 1 million users worldwide.

Finally, we want to emphasize that the use of ML surrogates is not intended to avoid or exclude HPC use in education. Instead, our vision is for ML surrogates to complement and supplement HPC-enabled simulations for education applications. Note also that the ML surrogate was designed using completed runs of HPC-enabled simulations. Without HPC, the time to generate the datasets to train the ML surrogate becomes prohibitively large [7], [8]. The use of ML surrogates contributes a novel way of teaching HPC topics by helping students develop intuition or "feel" for the physical system behavior through rapid exploration and visualization of variations in output quantities with changes in inputs, before they use HPC to solve specific problems.

ACKNOWLEDGMENT

References

[1] D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed. Academic Press, 2001.

[2] S. C. Glotzer, "Assembly engineering: Materials design for the 21st century (2013 pv danckwerts lecture)," *Chemical Engineering Science*, vol. 121, pp. 3–9, 2015.

[3] N. E. Brunk, M. Uchida, B. Lee, M. Fukuto, L. Yang, T. Douglas, and V. Jadhao, "Linker-mediated assembly of virus-like particles into ordered arrays via electrostatic control," *ACS Applied Bio Mat.*, vol. 2, no. 5, pp. 2192–2201, 2019.

[4] Y. Jing, V. Jadhao, J. W. Zwanikken, and M. Olvera de la Cruz, "Ionic structure in liquids confined by dielectric interfaces," *The Journal of chemical physics*, vol. 143, no. 19, p. 194508, 2015.

[5] V. Jadhao and M. O. Robbins, "Rheological properties of liquids under conditions of elastohydrodynamic lubrication," *Tribology Letters*, vol. 67, no. 3, p. 66, 2019. [Online]. Available: https://doi.org/10.1007/s11249-019-1178-3

[6] J. Kadupitiya, G. C. Fox, and V. Jadhao, "Machine learning for performance enhancement of molecular dynamics simulations," in *International Conference on Computational Science*, 2019, pp. 116–130. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-22741-8_9

[7] J. Kadupitiya, F. Sun, G. Fox, and V. Jadhao, "Machine learning surrogates for molecular dynamics simulations of soft materials," *Journal of Computational Science*, p. 101107, 2020.

[8] J. C. Kadupitige, G. Fox, and V. Jadhao, "Machine learning for auto-tuning of simulation parameters in car-parrinello molecular dynamics," in *APS Meeting Abstracts*, 2019.

[9] R. Tanaka, R. F. da Silva, and H. Casanova, "Teaching parallel and distributed computing concepts in simulation with wrench." in *EduHPC@ SC*, 2019, pp. 1–9.

[10] S. Srivastava, M. Smith, A. Ghimire, and S. Gao, "Assessing the integration of parallel and distributed computing in early undergraduate computer science curriculum using unplugged activities," in *2019 IEEE/ACM Workshop on Education for High-Performance Computing (EduHPC)*. IEEE, 2019, pp. 17–24.

[11] M. Spellings and S. C. Glotzer, "Machine learning for crystal identification and discovery," *AIChE Journal*, vol. 64, no. 6, pp. 2198–2206, 2018.

[12] S. S. Schoenholz, "Combining machine learning and physics to understand glassy systems," *Journal of Physics: Conference Series*, vol. 1036, no. 1, p. 012021, 2018.

[13] A. L. Ferguson, "Machine learning and data science in soft materials engineering," *Journal of Physics: Condensed Matter*, vol. 30, no. 4, p. 043002, 2017.

[14] K. Ch'ng, J. Carrasquilla, R. G. Melko, and E. Khatami, "Machine learning phases of strongly correlated fermions," *Phys. Rev. X*, vol. 7, p. 031038, Aug 2017.

[15] J. Kadupitiya, G. C. Fox, and V. Jadhao, "Machine learning for parameter auto-tuning in molecular dynamics simulations: Efficient dynamics of ions near polarizable nanoparticles," *Indiana University, Nov*, 2018.

[16] G. Fox, J. A. Glazier, J. Kadupitiya, V. Jadhao *et al.*, "Learning everywhere: Pervasive machine learning for effective high-performance computation," in *IEEE IPDPS Workshops*, 2019, pp. 422–429. [Online]. Available: https://doi.org/10.1109/IPDPSW.2019.00081

[17] J. Wang, S. Olsson, C. Wehmeyer, A. Perez, N. E. Charron, G. De Fabritiis, F. Noe, and C. Clementi, "Machine learning of coarse-grained molecular dynamics force fields," *ACS central science*, 2019.

[18] F. Häse, I. Fdez. Galván, A. Aspuru-Guzik, R. Lindh, and M. Vacher, "How machine learning can assist the interpretation of ab initio molecular dynamics simulations and conceptual understanding of chemistry," *Chem. Sci.*, vol. 10, pp. 2298–2307, 2019. [Online]. Available: http://dx.doi.org/10.1039/C8SC04516J

[19] Y. Sun, R. F. DeJaco, and J. I. Siepmann, "Deep neural network learning of complex binary sorption equilibria from molecular simulation data," *Chemical science*, vol. 10, no. 16, pp. 4377–4388, 2019.

[20] M. Kasim, D. Watson-Parris, L. Deaconu, S. Oliver, P. Hatfield, D. Froula, G. Gregori, M. Jarvis, S. Khatiwala, J. Korenaga *et al.*, "Up to two billion times acceleration of scientific simulations with deep neural architecture search," *arXiv preprint arXiv:2001.08055*, 2020.

[21] J. Kadupitiya, G. C. Fox, and V. Jadhao, "Simulating molecular dynamics with large timesteps using recurrent neural networks," *arXiv preprint arXiv:2004.06493*, 2020.

[22] G. Klimeck, M. McLennan, S. P. Brophy, G. B. A. III, and M. S. Lundstrom, "nanohub.org: Advancing education and research in nanotechnology," *Computing in Science Engineering*, vol. 10, no. 5, pp. 17–23, Sept 2008.

[23] N. E. Brunk and V. Jadhao, "Computational studies of shape control of charged deformable nanocontainers," *Journal of*

Materials Chemistry B, vol. 7, p. 6370, 2019. [Online]. Available: http://dx.doi.org/10.1039/C9TB01003C

[24] V. Jadhao, Z. Yao, C. K. Thomas, and M. Olvera de la Cruz, "Coulomb energy of uniformly charged spheroidal shell systems," *Physical Review E*, vol. 91, no. 3, p. 032305, 2015.

[25] K. Kadupitiya, N. Anousheh, S. Marru, G. C. Fox, and V. Jadhao, "Ions in nanoconfinement," Dec 2017, nanoHUB. [Online]. Available: https://nanohub.org/resources/nanoconfinement

[26] J. Kadupitiya, N. Brunk, S. Ali, G. C. Fox, and V. Jadhao, "Nanosphere electrostatics lab," May 2018, nanoHUB. [Online]. Available: https://nanohub.org/resources/nselectrostatic

[27] N. Brunk, J. Kadupitiya, M. Uchida, T. Douglas, and V. Jadhao, "Nanoparticle assembly lab," January 2019, nanoHUB. [Online]. Available: https://nanohub.org/resources/npassemblylab

[28] J. Kadupitiya, N. Brunk, and V. Jadhao, "Nanoparticle shape lab," January 2020, nanoHUB. [Online]. Available: https://nanohub.org/resources/npshapelab

[29] L. Nilsson, J. Kadupitiya, and V. Jadhao, "Polyvalent nanoparticle binding simulator," Apr 2019, nanoHUB. [Online]. Available: https://nanohub.org/resources/nanobind

[30] ——, "Souffle: Virus capsid assembly lab," Apr 2020, nanoHUB. [Online]. Available: https://nanohub.org/resources/capsidsouffle

[31] V. Jadhao, "Nanoscale simulations and engineering applications: Applications - self-assembly in nanoconfinement," Feb 2019. [Online]. Available: https://nanohub.org/resources/29671

[32] ——, "Shape changing nanocontainers tutorial," Apr 2020. [Online]. Available: https://nanohub.org/resources/33259

[33] F. J. Solis, V. Jadhao, and M. Olvera de la Cruz, "Generating true minima in constrained variational formulations via modified lagrange multipliers," *Physical Review E*, vol. 88, no. 5, p. 053306, 2013.

[34] S. Plimpton, "Fast parallel algorithms for short-range molecular dynamics," *Journal of Computational Physics*, vol. 117, no. 1, pp. 1 – 19, 1995.

[35] F. Chollet *et al.*, "Keras," 2015.

[36] L. Buitinck *et al.*, "Api design for machine learning software: experiences from the scikit-learn project," *arXiv:1309.0238*, 2013.

[37] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning." in *OSDI*, vol. 16, 2016, pp. 265–283.

[38] D. Marchant, C.-J. Johnsen, B. Vinter, and K. Skovhede, "Teaching concurrent and distributed programming with concepts over mathematical proofs," in *2019 Ieee/acm Workshop on Education for High-performance Computing (eduhpc)*. IEEE, 2019, pp. 49–57.

[39] A. Shoker, "Successful systems in production graduate teaching," in *2019 IEEE/ACM Workshop on Education for High-Performance Computing (EduHPC)*. IEEE, 2019, pp. 42–48.

[40] A. González-Escribano, V. Lara-Mongil, E. Rodriguez-Gutiez, and Y. Torres, "Toward improving collaborative behaviour during competitive programming assignments." in *EduHPC@ SC*, 2019, pp. 68–74.

[41] R. Carratalá-Sáez, S. Iserte, and S. Catalán, "Teaching on demand: an hpc experience." in *EduHPC@ SC*, 2019, pp. 32–41.

[42] A. Qasem, "A gentle introduction to heterogeneous computing for cs1 students," in *2019 IEEE/ACM Workshop on Education for High-Performance Computing (EduHPC)*. IEEE, 2019, pp. 10–16.

[43] J. Miller and M. Arenaz, "Measuring the impact of hpc training," in *2019 IEEE/ACM Workshop on Education for High-Performance Computing (EduHPC)*. IEEE, 2019, pp. 58–67.

[44] V. Jadhao and N. Makri, "Iterative monte carlo with bead-adapted sampling for complex-time correlation functions," *The Journal of chemical physics*, vol. 132, no. 10, p. 104110, 2010.

[45] ——, "Iterative monte carlo path integral with optimal grids from whole-necklace sampling," *The Journal of chemical physics*, vol. 133, no. 11, p. 114105, 2010.