Efficient Learning of Accurate Surrogates for Simulations of Complex Systems

A. Diaw,[1,*] M. McKerns,[1] I. Sagert,[1] L. G. Stanton,[2] and M. S. Murillo[3]

[1] *Los Alamos National Laboratory, Los Alamos, New Mexico 87545*
[2] *Department of Mathematics and Statistics, San José State University, San José, California 95192*
[3] *Department of Computational Mathematics, Science and Engineering, Michigan State University, East Lansing, Michigan 48824*

(Dated: October 26, 2021)

Estimation methods, such as interpolation and machine learning, are commonly used to build computationally inexpensive surrogates for physical models. Using an estimator to build a surrogate generally is a two-step process. First, a surrogate is fit by "training" on a subset of the existing data; then, the surrogate's performance is "tested" on the remaining (holdout) data. It is generally assumed that enough representative data was sampled for training so that if any further new data is encountered, the surrogate will perform approximately as well as it did against the original holdout data. The assumption is often poor, especially when data is sparse or the model is nonlinear. Worse still, when new data exposes a surrogate as a poor approximation of the model, there is often not a well-defined process to update the surrogate. As we are interested in finding a surrogate that provides valid predictions of any potential future model evaluations, we propose an online learning method that can leverage optimizer-driven sampling to ensure the critical points of the model are included in the training data. After any new model evaluations, surrogates are tested and "retrained" (updated) if the "score" drops below a validity threshold. We first assess the performance of our method on benchmark functions and find that optimizer-directed sampling generally outperforms traditional sampling methods both in terms of efficiency and accuracy in critical regions, even when a scoring metric that favors overall accuracy is used. Then, we apply our method to nuclear matter and neutral and charged liquids (including mixtures), demonstrating that highly accurate surrogates for the nuclear equation of state and radial distribution function can be reliably auto-generated as calculated by molecular dynamics simulations using a minimum of model evaluations.

MAIN

Throughout the diverse range of science and engineering applications, there is a growing desire to develop inexpensive surrogates for complex physical systems, where learned surrogates come with some guarantee in their ability to reliably predict the behavior of the system. Specifically, there is a strategic need for tools to be developed that can robustly forecast the behavior of complex physical systems, where data may be high-dimensional, noisy, or sparse, and models of the system may be time-dependent or include uncertainty.

For example, in material science[1], large-scale macroscopic simulations rely on closure information based on microscopic models that correctly reflect essential microphysical processes[2–5]. A fundamental challenge is to produce macroscopic simulations for a wide variety of phenomena, including phase transitions, complex mixtures, and shock waves, that are predictive at a level of confidence similar to that provided by high-fidelity microscopic simulations[6]. A significant amount of data generated from expensive microscopic models, such as molecular dynamics[6–8] or Monte Carlo[9,10] simulations, is essential in the production of statistically valid simulations of complex macroscopic phenomena. Unfortunately, generating sufficient data often requires a prohibitively large number of expensive microscopic simulations, and thus can quickly become impractical. This creates a significant roadblock to materials discovery and design, and the robust prediction of materials properties[11,12].

Bottlenecks due to prohibitively expensive simulations are not unique to materials science and are especially prevalent anywhere that robust prediction relies on multi-scale phenomena. Climate science, quantum information science, and, more generally, the automated control of instrumentation, all have a need for alternatives to using expensive simulations to make robust predictions[13–16]. Researchers have begun to turn to the learning of surrogates as brute-force computational approaches generally become intractable in large or complex systems. In fact, the generation of surrogates, through interpolation or machine learning, holds great promise in each of these cases to overcome the roadblocks to new science.

Recently, we used active learning to generate surrogates of fine-scale materials response[17,18], while Roehm et al.[19] used kriging to construct surrogates of stress fields and communicate the results to a fine-scale code that solves the macro-scale conservation laws for elastodynamics. Noack et al.[16] used a similar kriging-based approach to construct surrogates for autonomous x-ray scattering experiments. None of the above approaches attempt to ensure surrogates are valid on future data, and thus provide no guarantee on the validity of the surrogate. Instead, whenever a learned surrogate is evaluated, an uncertainty metric is also evaluated to determine if and where new fine-scale simulations should be launched. Noack, for example, uses a genetic algorithm to find

* diaw@lanl.gov

the maximum of the variances for each measured data point, then draws new samples from a distribution localized around the solved maximum. Passive approaches to validity such as these assume the expensive model will always be available and may have performance significantly hampered by the need to frequently request new expensive model evaluations throughout the life of the surrogates.

In the remainder of this paper, we present a new online learning methodology to efficiently learn surrogates that are *asymptotically* valid with respect to any future data. We use "asymptotically" valid to indicate that while we don't have a formal proof of validity versus future data, there is strong evidence to support our claim of at least approximate validity with respect to future data under some light conditions. More specifically, we conjecture that the minimum data set necessary to produce a highly accurate surrogate is one composed of model evaluations at all critical points on the model's response surface. Our conjecture comes with the condition that the selected class of surrogates has enough flexibility to reproduce the model accurately. Hence, it is beneficial to use a radial basis function (RBF)[20] as the estimator when training a surrogate for the model's response surface. The utility of RBFs arises from their universal function approximation capabilities[21], and their connection to single hidden-layer feed-forward neural networks (NN) with non-sigmoidal nonlinearities[22]. While a multilayer perceptron (MLP)[23] or another similar NN estimator should also be sufficient, we will use RBF interpolation in this paper, as RBFs are generally more efficient than MLP when using online learning[22].

Our methodology has three key components: a sampling strategy to generate new training and test data, a learning strategy to generate candidate surrogates from the training data, and a validation metric to evaluate candidate surrogates against the test data. For implementation, we leverage *mystic*[24,25], an open-source optimization, learning, and uncertainty quantification toolkit. *mystic* has over a decade of use in the optimization of complex models, including using uncertainty metrics to optimally improve model accuracy and robustness[26–30], and to increase the statistical robustness of surrogates[31–33]. Recent developments have included highly-configurable sampling strategies[34], which we will use in combination with online learning to train surrogates for accuracy with respect to all future data.

In this paper, we will primarily focus on how the choice of sampling strategy impacts efficiency in producing an asymptotically valid surrogate, with validity defined by the evolution of the surrogate test score. We will compare "optimizer-directed" sampling versus more traditional random sampling, in the efficiency of learning the response surface, where, fundamentally, both approaches utilize an identical first draw. In optimizer-directed sampling, the second draw uses first draw members as starting points for optimizers, as opposed to repeating the same probability sampling used in the first draw. This approach follows a snowball sampling of points for each first draw member, with the corresponding optimizer directing the sampling toward a critical point on the response surface. When an optimizer's termination condition is met, probability sampling is used as in the first draw to generate a new starting point for a new optimizer and again continues until termination.

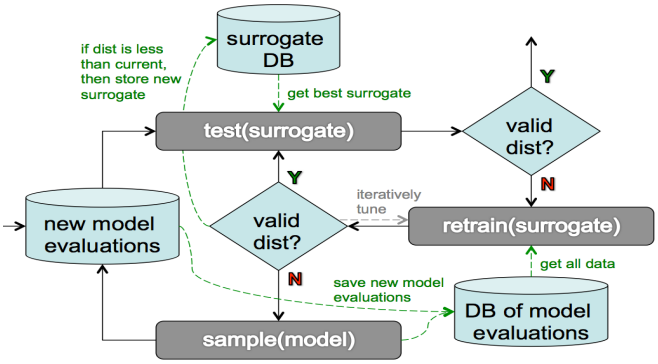

FIG. 1. Automated generation of computationally inexpensive surrogates for models of complex physical systems. When new model evaluations occur, the corresponding surrogate is retrieved and evaluated for the same data (see Method E). If the surrogate is deemed to still be valid after testing, execution stops. Otherwise, the surrogate is updated by retraining against the database of stored model evaluations, where the surrogate is validated with a fine-tuning of surrogate hyperparameters against a quality metric. If iterative retraining improves the surrogate, it is saved. Otherwise, we sample new model evaluations to generate new data. The process repeats until testing produces a valid surrogate.

THE ALGORITHM

We present our methodology for producing a valid surrogate for data due to existing model evaluations and then describe how to extend surrogate validity to future data. We will discuss each of the three key components of the methodology in detail, indicating where and how each component can be customized for a given use case. This section will serve as a reference for the specific configuration options selected for each case study in Section .

Surrogate Validity. A common goal in building a surrogate for some expensive process in complex systems is to provide a guarantee of the quality of the candidate surrogate. Generally, the validation of a surrogate against future real-world data is a laborious manual process that lacks a well-defined notion of a solution. The standard approach requires significant human effort in cleaning and reducing data, as well as selecting an estimator to generate candidate surrogates, "training" a candidate surrogate on a subset of the existing data, and "testing" the candidate surrogate on the remaining (hold-out) data. Validation of a surrogate with regard to future

data is often left as a post-hoc boolean decision involving relevant constraining information for each special use case. It is also generally assumed that enough representative data was sampled for training, so that if any further new data is encountered, the surrogate will perform approximately as well as it did against the original holdout data. The assumption that a representative amount of data has been collected is often poor, especially when data is sparse or the model is nonlinear. Further, when new data invalidates a surrogate, there is often not a well-defined process to update the surrogate.

Our general procedure for producing a valid surrogate for an expensive model is shown in Fig. 1. The procedure is iterative and includes explicit validation and update mechanisms. To simplify computational complexity, we first link the model to a database (as in Method A); thus, when the model is evaluated, the model's inputs and output are automatically stored. The database of model evaluations will be used later to train candidate surrogates (as in Method C). When the model is evaluated, the corresponding surrogate is retrieved from the surrogate database (as in Method E) and tested for validity (as in Method D). If no stored surrogate exists, then we skip testing and proceed directly to learning a candidate surrogate. We define validity when testing a surrogate as

$$\textit{test}(\Delta) \text{ is true} \tag{1}$$

where *test* is a function of the graphical distance, Δ

$$\begin{aligned} \Delta_y &= \inf_{\mathbf{x}\in\mathcal{X}} \left|\hat{y}(\mathbf{x}|\xi) - y\right| + \Delta_x, \\ \Delta_x &= \left|\mathbf{x} - \mathbf{x}'\right| \text{ or } 0 \end{aligned} \tag{2}$$

with $(\mathbf{x}', y)$ a point in the database of model evaluations, and $\mathcal{X}$ the set of all valid inputs $\mathbf{x}$ for the surrogate $\hat{y}$ with hyperparameters ξ. Δ_x and Δ_y are, respectively, the pointwise Δ for $(\mathbf{x}', y)$. If $\Delta_x = 0$, we ignore the distance of the inputs, and Δ_y is the minimum vertical distance of point y from the surrogate. We will define *test* on a per case basis, in Section .

If Eq. (1) deems the surrogate to be valid, execution stops; otherwise, we update the surrogate by training against the database of stored model evaluations (as in Method C). We define validity when training a surrogate similar to Eq. (1), but with the function *train* replacing *test*. We train the surrogate in terms of a quality metric, which is typically a distance such as $\delta = \sum_y \Delta_y$, or more generally

$$\delta = \textit{metric}(\hat{y}(\mathbf{x}|\xi),\ \textit{data}) \tag{3}$$

with *metric* a distance function between the surrogate and all model evaluations, *data* (as discussed in Method D). If, after training, a surrogate has a smaller δ compared to the current best surrogate, then we store the updated surrogate in the surrogate database and keep attempting to improve the surrogate until *train* is satisfied. If training ultimately fails to produce a valid surrogate, we use a sampler to generate model evaluations at new $(\mathbf{x}', y)$, and the process restarts.

Our general procedure for producing a valid surrogate is extended for asymptotic validity by adding a validity convergence condition

$$\textit{converged}(\Delta) \text{ is true} \tag{4}$$

to be called after the surrogate is deemed *test* valid, as in Eq. (1). Thus, instead of stopping execution when the surrogate is *test* valid, *test* valid merely completes an iteration. If not *converged*, we trigger a new iteration by sampling new data and continue to iterate until the surrogate validity has *converged*. This iterative procedure is more likely to generate a surrogate that is valid for all future data, when Eq. (4) requires some form of convergence behavior for *test* over several iterations. When the database of model evaluations is sparsely populated, we expect that any new data will likely trigger a surrogate update. Note that adding new data to the database does not ensure that a surrogate will become *more* valid; however, our conjecture is that as we sample more of the critical points on the model's response surface, our ability to learn a valid surrogate improves. To reduce the number of sampling iterations required, we can increase the number of samplers used in each iteration. We will discuss different sampling strategies in Section , highlighting the impact of the strategy on the efficiency of the calculation.

Learning Strategy. We are interested in finding a surrogate that is *asymptotically* valid against any future data, thus we propose an iterative *online* learning method that uses sampling to help ensure the critical points of the model are included in the training data. Online learning is a promising approach for the construction of reliably predictive surrogates, as recent studies[14,35] have demonstrated online learning can find non-intuitive parameter trajectories that significantly outperform the best results of a human operator in optimizing a complex system, and do so in much less time. Online learning provides a mechanism to update learned surrogates as new data is obtained. Our procedure is online, as a sampler can request new model evaluations on-the-fly, which populate to a database, and our surrogate is updated by querying the database and training on the stored model evaluations. A surrogate is tested when new data is obtained and is retrained (i.e. updated) whenever testing invalidates the current surrogate. Online approaches can be passive or can include strategies for requesting new data that, for example, have the goal of efficiently reducing the expected error between the measured data and predictions from the surrogate[32]. To this end, we will use online learning to find a surrogate that will reliably predict the model's response even where data has not yet been collected.

Online learning is greatly facilitated by automation of the learning process. Our general procedure for automating the production of a valid surrogate is shown in Fig.

1, and is extended to asymptotic validity as described in Section . As mentioned earlier, we will use an RBF to generate our surrogates, where we leverage *mystic*, as in Method C, to help with the automation and quality of the surrogate produced.

Let us assume $y(\mathbf{x})$ is an arbitrary function of vector $\mathbf{x}$ represented on a subset of $\mathbb{R}^n$, and that the value of y at input vectors $\mathbf{x}^j$ $(j = 1, \ldots, N)$ are the known *data* points stored in a database of model evaluations. We seek to find a surrogate $\hat{y}(\mathbf{x})$ with the lowest possible number of the evaluations[36–38] satisfying Eq. (3). We use Eq. (3), with $\mathbf{x}' = \mathbf{x}$, as opposed to

$$\hat{y}(\mathbf{x}^j) = y(\mathbf{x}^j) \text{ for all } j = 1, \ldots, N \tag{5}$$

as we allow our interpolated surrogates to deviate from the data slightly, due to the use of `smooth` and `noise` (see Method C). Using a RBF $\phi(r)$, the interpolated function can be written as:

$$\hat{y}(\mathbf{x}) = \sum_{j=1}^{N} \beta_j \phi(d(\mathbf{x}, \mathbf{x}^j)), \tag{6}$$

where β_j are coefficients to be determined, and $d(\mathbf{x}, \mathbf{x}^j)$ is a distance function similar to Eq. (3). If we choose $d(\mathbf{x}, \mathbf{x}^j) = ||\mathbf{x} - \mathbf{x}^j||$ as the Euclidean distance between an arbitrary vector $\mathbf{x}$ and $\mathbf{x}^j$, the values of the coefficient vector $\boldsymbol{\beta} = [\beta_1, \beta_2, \cdots, \beta_N]^T$ are determined by solving the linear system, $\mathbf{M}\boldsymbol{\beta} = \mathbf{Y}$, where $\mathbf{M}$ is an $N \times N$ symmetric matrix with elements $M_{ij} = \phi(||\mathbf{x}^i - \mathbf{x}^j||)$, and $\mathbf{Y} = [y(\mathbf{x}^1), y(\mathbf{x}^2), \ldots, y(\mathbf{x}^N)]^T$. In this work, we use `thin_plate` ($\phi = r^2 \ln(r)$) RBF to interpolate the data. To prevent issues due to singular matrix $\mathbf{M}$, and to provide some randomness in each learned surrogate, we add a very small amount of Gaussian noise to the input data.

Sampling Strategy. Sampling is an integral part of our online learning workflow and is used to generate new data points $(\mathbf{x}', y)$ that help inform the learning algorithm whenever training fails to produce a valid surrogate. As the goal is an asymptotically valid surrogate, we also use sampling to kick-start a new iteration after Eq. (1) deems the current iteration's surrogate to be valid (see Fig. 1). While the sampling and learning components of our workflow are fundamentally independent, and are able to run asynchronously, they are linked through the database of stored model evaluations. The data points generated by the sampler are populated to the database (as in Method A), while the learning algorithm always uses the data contained in the database at the time a new training is requested. If there were no concerns about minimizing the number of model evaluations, we could have samplers run continuously, feeding model evaluations into the database. However, we explicitly include sampling as part of the iterative workflow, as described above, in an attempt to minimize the number of model evaluations.

We conjectured that a learned surrogate is guaranteed to be valid for all future data, given the training data, at a minimum, includes all of the critical points of the response surface $y(\mathbf{x})$. Thus, we postulate that a sampling strategy that uses *optimizer-directed* sampling will be most efficient in discovering all the critical points of $y(\mathbf{x})$. We distinguish *optimizer-directed* sampling from *traditional* sampling in that optimizer-directed sampling uses an optimizer to direct the sampling toward a goal, while traditional methods, such as simple random sampling, generally ignore the response of the function $y(\mathbf{x})$. The utility of simple random sampling is that all the samples will draw (with replacement) from a distribution, and thus all sample points can be chosen simultaneously, and subsequently $y(\mathbf{x})$ can be evaluated in parallel for all points drawn in the sampling. An optimizer-directed approach uses traditional sampling to generate samples for the first draw, then uses each first draw member as a starting point for an optimizer that will direct the sampling of the second and subsequent draws toward a critical point on the response surface. When an optimizer's termination condition is met, traditional sampling is again used to generate a new starting point for a new optimizer, which then proceeds to termination as above. Thus, while an optimizer-directed strategy may be less efficient in generating new data points, it should be more efficient at finding the critical points of the response surface, and thus we expect that it should also be the preferred strategy when a surrogate is required to be asymptotically valid.

Our sampling approach is implemented in *mystic* in `mystic.samplers`, where we have a choice of three first draw strategies: (1) "*random*", simple random sampling from a distribution, with replacement, (2) "*sparsity*", systematic random sampling with no replacement, where new draws occur at a maximum distance from all existing data, or (3) "*lattice*", sampling on a grid, with each drawing located at the center of a bin in gridded parameter space. The sampling algorithm can be further configured, as described in Method B, where as a first approximation to solving for the critical points, we will use optimizers to search for all minima and maxima of the response surface. We will test the performance of different sampling algorithms focusing on the capability and performance of optimizer-directed sampling versus traditional methods, choices of different first draw strategies, choices of different optimizers, the number of optimizers used in the ensemble, and the number of samples required before the next convergence check using Eq. (4) (i.e. the number of samples to 'warm start' each iteration).

NUMERICAL EVALUATION

In this section, we will assess the performance of different sampling strategies against benchmark functions. We then apply our methodology to finding accurate surrogates for two types of physical models: EOS calculations of dense nuclear matter and RDFs from expensive MD simulations. For the latter type of problem re-

garding RDFs, we will consider three different systems with the increasing complexity of the parameter space: the one-component plasma (OCP) model with a two-parameter space, the Lennard-Jones (LJ) fluid with a three-parameter space, and finally an extension of the OCP, the binary-ionic mixture (BIM) model with a five parameter space.

Benchmark Functions

We begin our case studies with several benchmark functions commonly used to test the performance of numerical optimization algorithms. We will first examine how sampling strategy impacts the efficiency and effectiveness of finding an asymptotically valid surrogate. Then, we will explore how optimizer configuration impacts the efficiency of generating an initial valid surrogate.

Sampling for Asymptotic Validity. We first contrast the ability of optimizer-directed sampling with random sampling in finding an accurate surrogate with regard to all future data. We use the workflow for asymptotic validity described in Section to learn a surrogate for the d-dimensional Rastrigin function[39]:

$$f(\mathbf{x}) = 10d + \sum_{i=1}^{d} \left[x_i^2 - 10\ \cos(2\pi x_i)\right] \tag{7}$$

with $d = 2$, bounded by $x_i \in [0, 10]$, where the 2-D Rastrigin's function is essentially a spherical function with a cosine modulation added to produce several regularly distributed local minima.

Our optimizer-directed sampler uses a "sparsity" sampling strategy with an ensemble of 16 `NelderMeadSimplexSolver` instances, configured as in Method B. We define our *test* for validity, in Eq. (1), as:

$$\text{ave}(\Delta_y) \leq tol_{ave}\ \wedge\ \max(\Delta_y) \leq tol_{max}, \tag{8}$$

where $tol_{ave} = 1 \cdot 10^{-5}$ and $tol_{max} = 1 \cdot 10^{-4}$. We use a graphical distance with $\Delta_x \neq 0$, and *data* defined as all existing model evaluations (i.e. prior plus newly sampled). We define *train* as in Eq. (8), again with $tol_{ave} = 1 \cdot 10^{-5}$ and $tol_{max} = 1 \cdot 10^{-4}$. We use a quality *metric* for training, defined by $\delta = \sum_y \Delta_y$. We define *converged*, in Eq. (4), as:

$$\Omega(M)\ \vee\ \max_y(\max_j(\text{ave}(\Delta_{y,j}))) \leq tol_{stop}\ , \tag{9}$$

with $\Omega(M)$ true when no new local extrema were found in the last $M = 3$ iterations, $tol_{stop} = 2 \cdot 10^{-4}$, $\Delta_{y,i}$ is the graphical distance to the *data* sampled in iteration i (i.e. no prior model evaluations), and j defined by the last $N = 3$ iterations $j \in [i - N + 1, ..., i]$. We also ensure that at least 1000 model evaluations have been performed per iteration, by setting $warm = 1000$. Additionally, we track the testing score for a single iteration i, defined as:

$$score = \text{ave}_y(\text{ave}(\Delta_{y,i}))\ , \tag{10}$$

although *score* is not used in terminating the calculation. We then repeat the calculation for $tol_{ave} = 1 \cdot 10^{-7}$, $tol_{max} = 1 \cdot 10^{-6}$, and $tol_{stop} = 2 \cdot 10^{-6}$, to assess the effects of tighter tolerances. We will refer to these tighter tolerance settings as "strict" and the prior tolerance settings as "loose".

We compare our results with pure systematic random sampling, using an ensemble of 500 points, for both strict and loose tolerances.

As can be seen in Figures 2 and 6, and in Table I, the test score for pure systematic random sampling converges yielding an excellent representation of truth faster than optimizer-directed sampling (when using the default optimizer configuration, and a metric based on the average surrogate misfit) for both strict and loose tolerances. As the Rastrigin function has shallow local extrema distributed uniformly across the response surface, it may be expected that a systematic random sampling strategy that efficiently covers input space is more performant than a strategy that attempts to pinpoint the extrema.

We also performed similar comparisons for the d-dimensional Rosenbrock function[40]:

$$f(\mathbf{x}) = \sum_{i=1}^{d-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 \tag{11}$$

with $d = \{2, 8\}$, d-dimensional Hartmann's function[41]:

$$f(\mathbf{x}) = -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{d} A_{ij}\left(x_j - P_{ij}\right)^2\right) \tag{12}$$

with $d = 6$, d-dimensional Michalewicz's function[42]:

$$f(\mathbf{x}) = -\sum_{i=1}^{d} \sin(x_i) \sin^{20}(ix_i^2/\pi) \tag{13}$$

with $d = 2$, and 2-dimensional Easom's function[43]:

$$f(\mathbf{x}) = -\cos(x_1)\cos(x_2)\, \mathrm{e}^{-(x_1-\pi)^2-(x_2-\pi)^2}. \tag{14}$$

The coefficients α_i, A_{ij} and P_{ij} for Hartmann's function can be found in Method G. The 2-D Rosenbrock function is a saddle with an inverted basin, where the global minima occur inside a long, narrow, flat parabolic valley in the inverted basin. The 8-D Rosenbrock function is the sum of seven coupled 2-D Rosenbrock functions, with a global minimum at $x_i = 1$ and a local minima near $x = [-1, 1, ..., 1]$. The 2-D Michalewicz's function generally evaluates to zero, however has long narrow channels that create local minima, and sharp dips at the intersections of the channels. The 2-D Easom's function is a unimodal function that evaluates to zero everywhere except in the region around a singular sharp well. The 6-D Hartmann's function is similar to Easom's function in that it generally evaluates to zero; however, it is multimodal and composed of several coupled intersecting sharp wells near the origin.

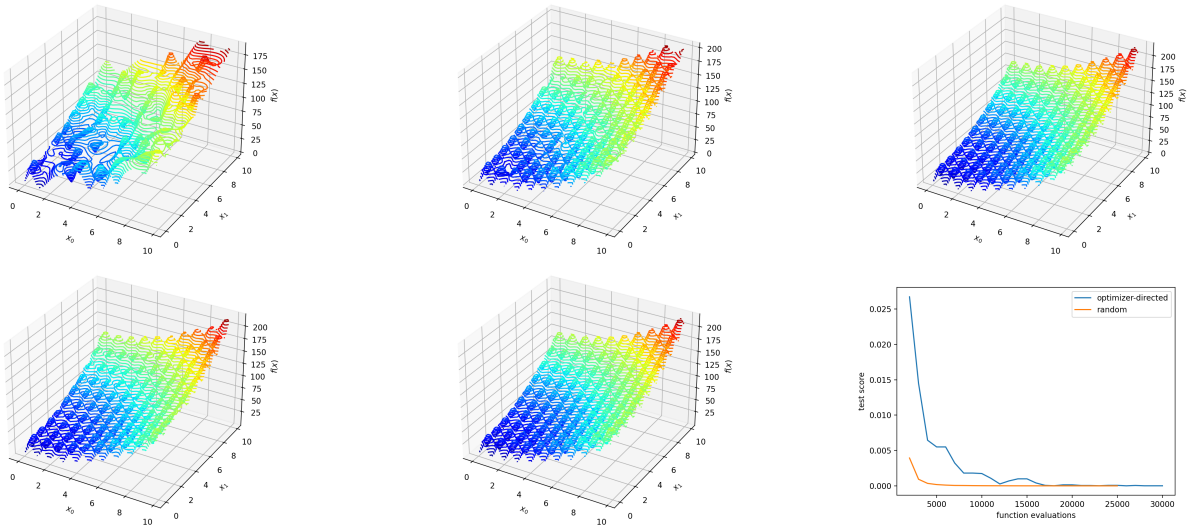

FIG. 2. Candidate surrogates for the 2-dimensional Rastrigin function, learned with a thin-plate RBF estimator using "sparsity" sampling, a "strict" tolerance, and a test metric for validity based on the average graphical distance between the learned surrogate and sampled data. Surrogates are plotted with inputs $x = (x_0, x_1)$ and output $z = f(x)$. Top row: Sampling using ensembles of 16 optimizers, after the initial, tenth, and final iteration. The final surrogate is visually identical to truth, and the surrogate reproduces all local extrema within the desired accuracy. Bottom row: Sampling using ensembles of 500 points, after the initial and tenth iteration. Bottom row, right: test score per sample. Note that the test score for pure systematic random sampling converges faster than optimizer-directed sampling, as may be expected for a metric based on the average surrogate misfit.

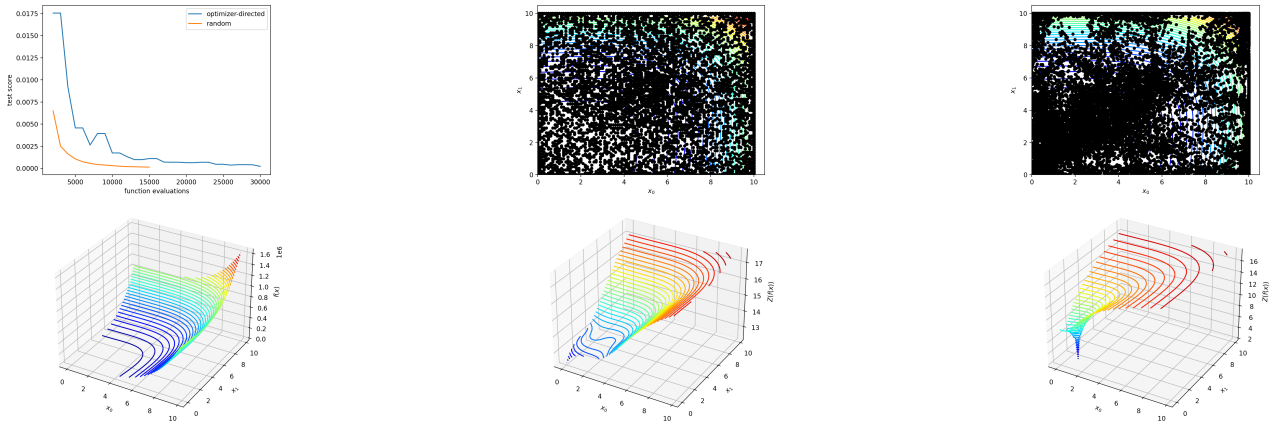

FIG. 3. Candidate surrogates for the 8-dimensional Rosenbrock function, learned with a thin-plate RBF estimator using "sparsity" sampling, a "loose" tolerance, and a test metric for validity based on the average graphical distance between the learned surrogate and sampled data. Surrogates are plotted with inputs $x = (x_0, x_1, 1, 1, 1, 1, 1, 1)$ and output $z = f(x)$ or $z = Z(f(x))$, where log-scaling $Z = log(4 \cdot f(x) + 1) + 2$ is used to better view the region around the global minimum. Top row, left: test score per sample. Top row, center: model evaluations sampled with the random sampling strategy. Top row, right: optimizer-directed sampling. Bottom row, left: surrogates produced with either sampling approach are visually identical to the truth. Bottom row, center: log-scaled view of surrogate from random sampling near the global minimum. Bottom row, right: log-scaled view of surrogate from optimizer-directed sampling near the global minimum, identical to the truth. Note that while pure systematic random sampling converges faster, optimizer-directed sampling provides a more accurate surrogate near the critical points.

The general form of the saddle in the 2-D Rosenbrock function is captured well with either systematic random sampling or optimizer-directed sampling, with the former again converging more quickly (as can be seen in Table I) for both strict and loose tolerances. However, upon closer examination of the long narrow channel that contains the global minimum (see Figure 5), we find that optimizer-directed sampling reproduces truth much more accurately in the region surrounding the global minimum. This is to be expected, as an optimizer-directed strategy

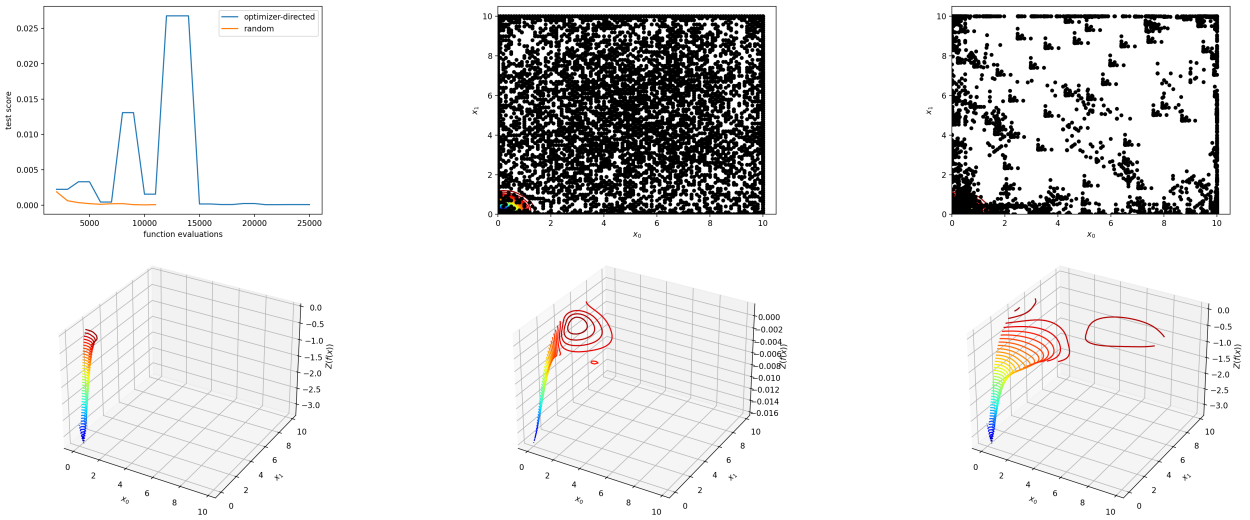

FIG. 4. Candidate surrogates for the 6-dimensional Hartmann function, learned with a thin-plate RBF estimator using "sparsity" sampling, a "loose" tolerance, and a test metric for validity based on the average graphical distance between the learned surrogate and sampled data. Surrogates are plotted with inputs $x = (x_0, x_1, 0.476874, 0.275332, 0.311652, 0.6573)$ and output $z = Z(f(x))$, where log-scaling $Z = log(4 \cdot f(x) + 1) + 2$ is used to better view the region around the global minimum. Top row, left: test score per sample. Note the spikes during convergence for optimizer-directed sampling, due to a 'dramatic' update to the surrogate. The 'dramatic' updates occur when sampling occurs in the region around $x_i = 2$, and better shape the transition from the flat region to the entry of the well. Top row, center: model evaluations sampled with the random sampling strategy. Top row, right: optimizer-directed sampling. Optimizer trajectories generally are short, and terminate quickly in the flat region, as seen by the triangle-shaped clusters in the top right panel. Bottom row, left: truth. Bottom row, center: surrogate from random sampling. Bottom row, right: surrogate from optimizer-directed sampling. Note that while optimizer-directed sampling produces a more accurate surrogate, neither reproduce truth.

should provide a higher density of sampling in the neighborhood of the single global minimum. We find similar behavior for both strict and loose tolerances, and for the 8-D Rosenbrock function tested with loose tolerances (see Figures 6 and 3).

The 2-D Easom's function and the 6-D Hartmann's function provide a similar challenge in that they evaluate to zero everywhere, except in the region around one or more sharp wells. Gradient-based optimizers do not navigate a flat surface well, so some inefficiency should be expected for optimizer-directed sampling for these functions. While again, optimizer-directed sampling (at the default optimizer configuration and for a metric based on the average surrogate misfit) converges less quickly, the difference is not as lopsided as one might anticipate. As seen in Figure 4, optimizer trajectories generally are short, and terminate quickly in the flat region; however, the region around a well has a higher density than random sampling. The result is that optimizer-directed sampling again produces a better representation of truth near the extrema. Note that for the 6-D Hartmann's function, optimizer-directed sampling does not capture the behavior of truth exactly around the rim of the well. This is due to our first-order simplification of the methodology searching for extrema, as opposed to all critical points of the response surface. Had we include the turning points in the search criteria, we expect the surrogate would do better at reproducing truth at the rim of the well.

The outlier of the study is the 2-D Michalewicz function. The function is generally flat, with several long narrow channels that have sharp dips at the intersections of the channels, and thus provide difficult features for both sampling approaches. We found that either approach visually reproduces truth, approximately, after $30,000$ model evaluations with loose tolerances. It is expected that for tighter tolerances, much more sampling is required to generate a surrogate for the 2-D Michalewicz function that reproduces the critical points with the quality observed for the other benchmark functions.

In summary, for all cases, systematic random sampling was found to converge more quickly to a surrogate that is valid for all future data. However, using optimizer-directed sampling, better ensures that the behavior at function extrema is reproduced. Note that in all cases, the optimizer used was a `NelderMeadSimplexSolver` in the default configuration (see Method F). We expect that less strict convergence requirements will reduce the number of evaluations required by the optimizer, potentially at the cost of some accuracy in the vicinity of the extrema. We will explore the impact of optimizer configuration in the next subsection.

Sampling for Training Validity. We now assess the impact on the efficiency of optimizer-directed sampling due to the configuration of the optimizer. Our optimizer-directed sampler uses a "lattice" sampling strategy with an ensemble of 40

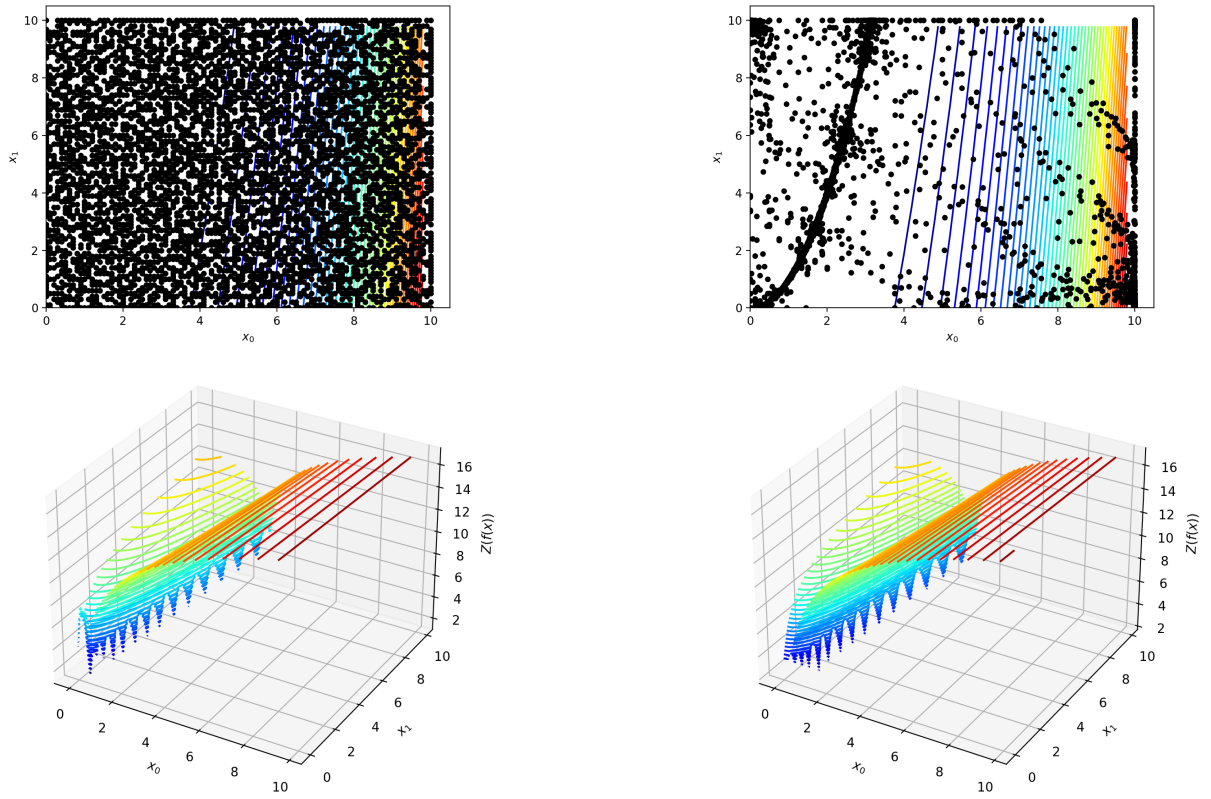

FIG. 5. Candidate surrogates for the 2-dimensional Rosenbrock function, learned with a thin-plate RBF estimator using "sparsity" sampling, a "loose" tolerance, and a test metric for validity based on the average graphical distance between the learned surrogate and sampled data. Surrogates are plotted with inputs $x = (x_0, x_1)$ and output $z = Z(f(x))$, where log-scaling $Z = log(4 \cdot f(x) + 1) + 2$ is used to better view the region around the global minimum. Top row, left: model evaluations sampled with the random sampling strategy. Top row, right: optimizer-directed sampling. Bottom row, left: log-scaled view of surrogate from random sampling. Bottom row, right: log-scaled view of surrogate from optimizer-directed sampling, which reproduces truth. Note convergence occurs quickly using either strategy, where the *converged* condition is met with no more than two iterations.

`NelderMeadSimplexSolver` instances, configured as in Method B. We define our *test* for validity, in Eq. (1), as:

$$\sum_y \Delta_y \leq tol_{sum} \ \wedge \ \max(\Delta_y) \leq tol_{max}, \qquad (15)$$

where $tol_{sum} = 1 \cdot 10^{-3}$ and $tol_{max} = 1 \cdot 10^{-6}$. We use a graphical distance with $\Delta_x \neq 0$, and *data* defined as all existing model evaluations (i.e. prior plus newly sampled). We define *train* as in Eq. (15), again with $tol_{sum} = 1 \cdot 10^{-3}$ and $tol_{max} = 1 \cdot 10^{-6}$. We use a quality *metric* for training, defined by $\delta = \sum_y \Delta_y$, and define *converged*, in Eq. (4), identically to *test* in Eq. (15).

In Table II, we present the average time to obtain validity (as defined above) for surrogates of several standard benchmark functions. Simulations were performed on the Darwin cluster at Los Alamos National Laboratory on a 36 core Skylake microarchitecture. We found for the default optimizer configurations (Method F), an ensemble of Nelder-Mead optimizers is more efficient, regardless of the dimensionality of the benchmark function. We surmise this is due to the Nelder-Mead solvers getting stuck in local extrema faster than those using Powell's method. As the goal here is for the ensemble of solvers to find as many local extrema as possible using the minimum number of function evaluations, Nelder-Mead appears to be the better choice. To dig further into the impact of optimizer configuration, we examined minimization of the 6-D Hartmann's function, Eq. (12), in the region bounded by $\mathbf{x} \in [-1, 1]$. The function has a single global minimum, $f(x) = -3.322$, that occurs at $x = \big[0.20169, 0.15001, 0.4768, 0.2753, 0.311, 0.6573\big]$. Figure 7 plots the convergence of (a) the inputs and (b) the output, for the 6-D Hartmann's function. The results depicted in Table II were obtained using a "lattice" sampling strategy with an ensemble of optimizers. Our ensembles were built with either a Nelder-Mead solver, or a Powell's Directional solver, in the default configuration (as shown in Method B). The former solver will stop when the absolute difference in both $\mathbf{x}$ and $f(\mathbf{x})$ over one iteration is less than 10^{-4}, while the latter solver will stop when the normalized absolute difference of $f(\mathbf{x})$ over

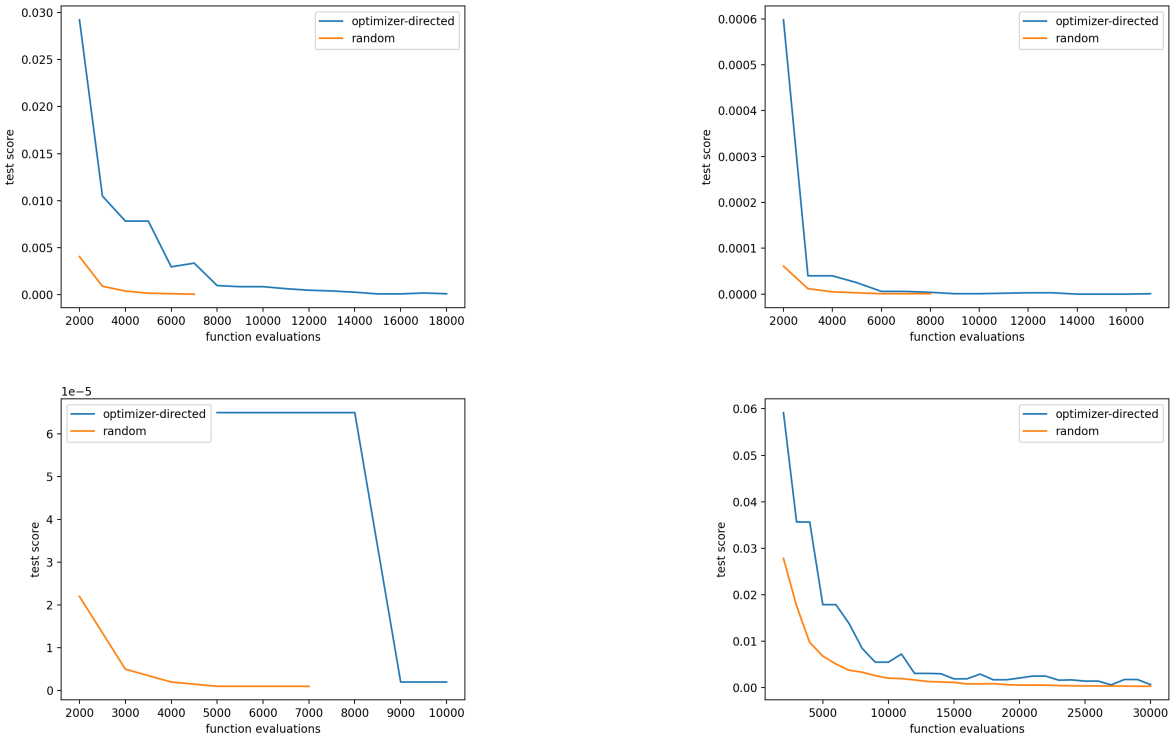

FIG. 6. Convergence of *test* score versus model evaluations for different benchmark functions learned with a thin-plate RBF estimator using "sparsity" sampling, and a test metric for validity based on the average graphical distance between the learned surrogate and sampled data. Top left: 2-dimensional Rastrigin function with "loose" tolerance. Top right: 2-dimensional Easom function with "strict" tolerance. Bottom left: 2-dimensional Rosenbrock function at strict tolerance. Bottom right: 2-dimensional Michalewicz function at loose tolerance. Note the surrogates on the top row reproduce truth well regardless of approach, as in Figure 2. Performance of either approach for the 2-D Rosenbrock is as occurs for the 8-D Rosenbrock, in Figure 3 Both random and optimizer-directed approaches do equally poorly with the 2-D Michalewicz.

2 iterations is less than 10^{-4}. In Figure 7, the cost appears to converge by the second iteration and remains unchanged for roughly ten subsequent iterations before the solver terminates. Hence, we should be able to reduce the number of function evaluations by roughly an order of magnitude with some tuning of the termination conditions (e.g. by setting `ftol` to 10^{-2}, and `xtol` to 10^{-2} for the Nelder-Mead solver). Tuning the optimizer termination conditions for each particular benchmark function will likely produce a significant drop (e.g. an order of magnitude) in the number of function evaluations reported for optimizer-directed sampling in Table II.

The following sections will test the ability to quickly produce a valid surrogate that reproduces relevant physical behavior in regions where traditional methods have difficulty producing similar results. We will use a larger ensemble of optimizers and test the accuracy at the end of a single iteration of our entire workflow. This does not guarantee the surrogate will be valid against all future data but will give us an idea of how quickly the surrogates can accurately reproduce physical effects near the critical points.

Equation of State with Phase Transition.

We are interested in building an accurate surrogate for an equation of state (EOS) for a dense nuclear matter that contains a phase transition (PT). Reliable hadronic models for nuclear matter exist up to baryon number densities n_b of about twice nuclear saturation density $n_0 \sim 0.16$ fm^{-3} and at asymptotically high densities of $n_b \gg 40 n_0$[44,45]. While at low densities and temperatures T, the nuclear matter is composed of neutrons and protons, for high values of n_b and T, it is expected to undergo a PT to a phase composed of deconfined quarks and gluons. This quark-hadron PT is studied experimentally in heavy-ion collisions on Earth and might be present in the interior of neutron stars[46]. However, there are large uncertainties regarding the critical temperatures and densities for the onset of the PT.

Numerical studies of matter during heavy-ion collisions, core-collapse supernovae, neutron star merger events and within the interior of neutron stars, require the usage of a nuclear EOS[47]. The most common approach to creating an EOS over a large density range is to select realistic hadronic and quark models and con-

Function	ndim	Random		Optimizer-directed	
		loose	tight	loose	tight
Easom	2	2000	8000	2939	17967
Rosenbrock	2	2000	7000	7317	12111
Rastrigin	2	7000	25000	18579	32308
Michalewicz	2	30000	–	30696	–
Hartmann	6	11000	–	26411	–
Rosenbrock	8	15000	–	31487	–

TABLE I. Number of evaluations required for several benchmark functions to reach tol_{stop} for both "loose" and "strict" tolerances, using a `SparsitySampler` with bounds $\mathbf{x} \in [0, 10]$. In all cases, systematic random sampling converges more quickly to a surrogate that is valid for all future data. Using optimizer-directed sampling, however, ensures that the function extrema are known. The optimizer used was a `NelderMeadSimplexSolver` at the default configuration. Less strict convergence requirements will reduce the number of evaluations required by the optimizer, potentially at the cost of some accuracy in the vicinity of the extrema.

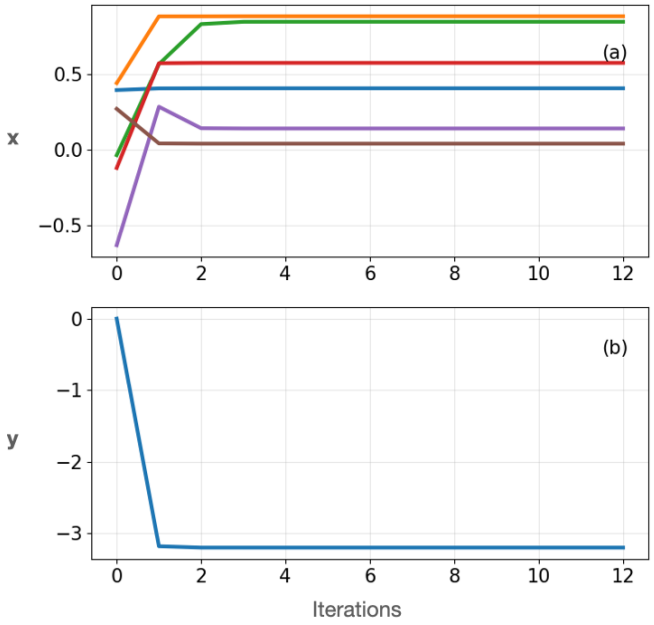

FIG. 7. Convergence of (a) the input parameters $\mathbf{x}$, and (b) output $y = f(\mathbf{x})$, for the 6-dimensional Hartmann function, Eq. (6), using a Nelder-Mead Solver in the default configuration, Method F. Note that convergence in $f(\mathbf{x})$ is reached within the first two iterations. Thus, with some tuning to the termination conditions, we expect an additional order of magnitude efficiency may be achieved. We found similar results using a Powell solver.

nect them either via a Maxwell or a Gibbs construction to describe the PT[48]. The Gibbs construction assumes that conservation laws are fulfilled globally in the quark-hadron mixed-phase, which results in pressure being a smooth function of the density. For the Maxwell construction, only baryon number is conserved globally, while other conservation laws like electric charge neutrality are fulfilled locally in the quark and hadronic phases.[48–50]. Neither of these constructions is currently ruled out, but the Maxwell construction usually leads to a more extreme behavior during the PT, leading to a pressure plateau in the so-called quark-hadron mixed phase.

Function	ndim	bounds	Function evaluations	
			Powell	Nelder-Mead
Ackley	2	[-1,1]	3746	1631
Branins	2	[-10,20]	2767	1007
Rosenbrock	3	[-3, 3]	4796	1733
Michalewicz	5	[0, 3]	12745	1116
Hartmann	6	[-1, 1]	11896	1393
Rosenbrock	8	[-6, 6]	18430	1185

TABLE II. Comparison of time to validity for several benchmark functions, using a `LatticeSampler` with an ensemble of 4 optimizers with the default configuration. We present the number of function evaluations required to find a valid surrogate, where *converged* and *test* are defined as in Eq (15), with $tol_{sum} = 10^{-3}$ and $tol_{max} = 10^{-6}$, and *train*, *data*, and *metric* defined as in Section . The sampler is configured to run until all of the optimizers in the ensemble have terminated.

For astrophysical simulations of core-collapse supernovae and neutron star mergers, nuclear matter EOS tables are used with thermodynamic quantities being functions of n_b, the proton fraction y_p and the temperature T[47]. The construction of these tables is time-intensive with many points in the (n_b, y_p, T) space, while their implementation in computational fluid dynamics (CFD) problems requires interpolation and inversion routines. A new approach to facilitate the usage of EOS tables is the construction of analytic fitting functions, which can then be included in the numerical simulations instead of an EOS table[51]. While promising, this method has yet to be shown to be able to accurately model extreme features like high-density PTs.

To demonstrate our framework, we implement a quark-hadron PT into a simple nucleonic model that is frequently used in astrophysics and nuclear physics[52,53]. Here, the nucleonic EOS is derived from the Skyrme-Hartree-Fock self-consistent mean-field model[54]. The energy of the system is determined as the expectation value of an effective nuclear Hamiltonian, which contains the zero-range Skyrme nuclear interaction[55]. For high-density neutron star interiors at zero or low temperatures, nuclear matter can be treated as degenerate and infinite with constant density. This greatly simplifies

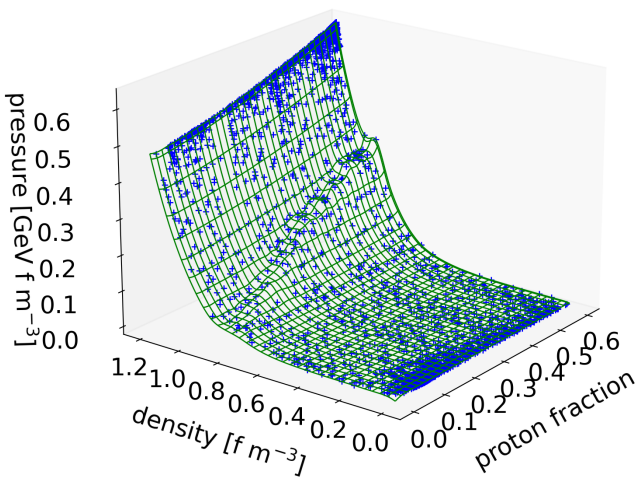

FIG. 8. Simulations (dots) and predicted values (surface) of the equation of state for quark matter. The initial search domain chosen as density $[fm^{-3}] \in [0.04, 1.2]$ and proton fraction $\in [0., 0.6]$ was sampled with a lattice sampler. We used 40 Nelder-Mead solvers and *test* validity defined as in Eq. (15) with $tol_{max} = 10^{-6}$, $tol_{sum} = 10^{-3}$, and *train*, *converged*, *data*, and *metric* as defined in Section . A valid surrogate for the EOS that correctly describes the plateau region was found after 7382 function evaluations.

the expression for the Skyrme energy density functional. In addition, the many-body state of the system can be expressed as a Slater determinant of uncorrelated plane wave states from lowest momentum up to the Fermi momentum. As a result, the energy per baryon of nuclear matter composed of neutrons and protons with densities ρ_n and ρ_p, respectively, can be written in some an analytic form as shown Method H.

Given the energy or energy per baryon, other thermodynamic properties, such as pressure, can then be determined by standard relations[54]. The PT is modeled by the Maxwell construction, while quark matter is described by the MIT Bag model, where quarks are non-interacting fermions with a negative confinement pressure, the so-called bag constant[50,56]. The latter model ensures that quarks are confined into neutrons and protons at low densities. Quark matter in our approach uses a bag constant of 170 MeV. It is composed of up, down, and strange quarks, where we assume that the masses of up and down quarks are negligible in comparison to their chemical potential, while the strange quark mass is set to 150 MeV. Although nucleonic and quark matter are given by simple models, our intention here is to demonstrate the ability of our framework to use expensive EOS data, even in the presence of a PT, to find a surrogate that can be directly used in high-fidelity CFD codes. EOS tables in astrophysical simulations usually have three degrees of freedom: n_b, y_p and T[47], but for simplicity, we will model a system where the pressure is a function of n_b and y_p only, and temperature effects are negligible, which is a reasonable assumption for systems such as neutron stars interiors.

We use our approach to find an accurate surrogate for the quark-hadron EOS for $0.04\ \mathrm{fm}^{-3} \leq n_b \leq 1.6\ \mathrm{fm}^{-3}$ and $0 \leq y_p \leq 0.6$. We used "lattice" sampling with an ensemble of 40 Nelder-Mead solvers at the default configuration, and a surrogate learned using a thin-plate RBF. Here, we defined *test* validity as in Eq. (15) with $tol_{max} = 10^{-6}$ and $tol_{sum} = 10^{-3}$, and *train*, *converged*, *data*, and *metric* as defined in Section . The results are plotted in Figure 8, which shows the entire n_b-y_p plane. The pressure plateau of the PT is clearly visible with critical density for the onset of the mixed phase moving to higher values for increasing proton fraction as discussed in[50]. Figures 9 (a)-(f) give a more detailed view of the EOS by showing the pressure profiles for fixed y_p and n_b, respectively. It can be seen from the figure that no systematic errors arise in the predicted values of the pressure with either proton fraction or density.

Note that the RBFs will reproduce the phase transition accurately if enough of the inflection points of the response surface are sampled. As the optimizers discover all the critical points in the surface (here, the points in the discontinuity), the smooth functions also begin to reproduce these discontinuities. We illustrated this point in Fig. 10.

Radial distribution functions

Here, we are interested in building accurate surrogates for radial distribution functions (RDF) for systems of neutral and charged liquids, where the data for the RDFs were computed with large-scale molecular dynamics (MD). RDF describes interparticle correlations within isotropic materials by averaging over relevant atomic or molecular coordinates; it is an important fluid characteristic, necessary for analyzing x-ray Thomson scattering experiments[57], investigating protein interactions with cell membranes[58], and examining shock-induced phase transitions[3]. In addition, the RDF plays a key role in perturbative fluids theories, as many aspects of thermodynamic properties in fluids can be expressed in terms of it. The RDF is also useful in investigating some inhomogeneous fluid structures[59], for example in dense plasmas, mixing phenomena[6], and enhancement of nuclear reaction rates[60]. Furthermore, recent developments in kinetic theory[61] and hydrodynamics[62] models for dense plasmas highlight the importance for RDF models. Because of the fundamental role of RDFs in our understanding of physical phenomena in fluids, it has been the focus of numerous studies.

The Lennard-Jones (LJ) model is among the most widely used models for neutral liquids, and numerous fits to its RDF have been given. Early work by Goldman[63] argues for as few parameters as possible for ease of use and to avoid overfitting to noise in the data; fits were

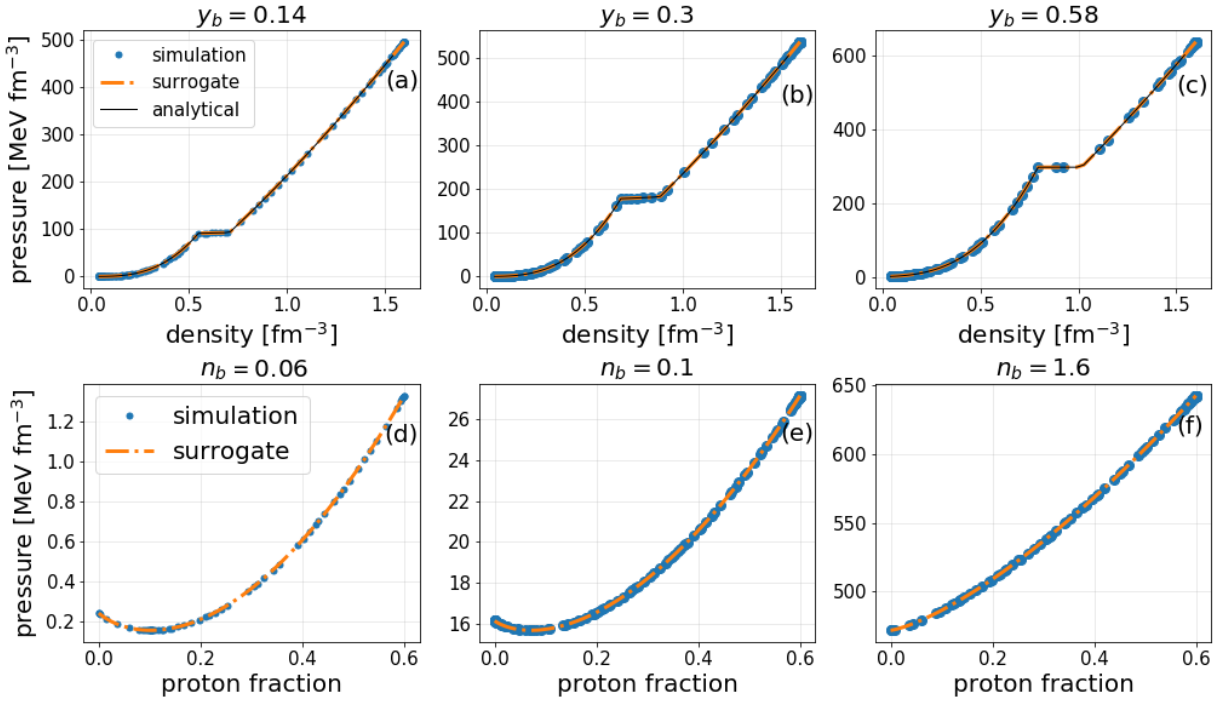

FIG. 9. Quark matter pressure as function of density and proton fraction. The dotted curves were obtained using a lattice sampler with 40 Nelder-Mead solvers and *test* validity defined as in Eq. (15) with $tol_{max} = 10^{-6}$, $tol_{sum} = 10^{-3}$, and *train*, *converged*, *data*, and *metric* as defined in Section . The dots are the results from a more expensive simulation and the black line corresponds to EOS analytical formula. We obtained a very good agreement between the simulations and the surrogates across parameter space.

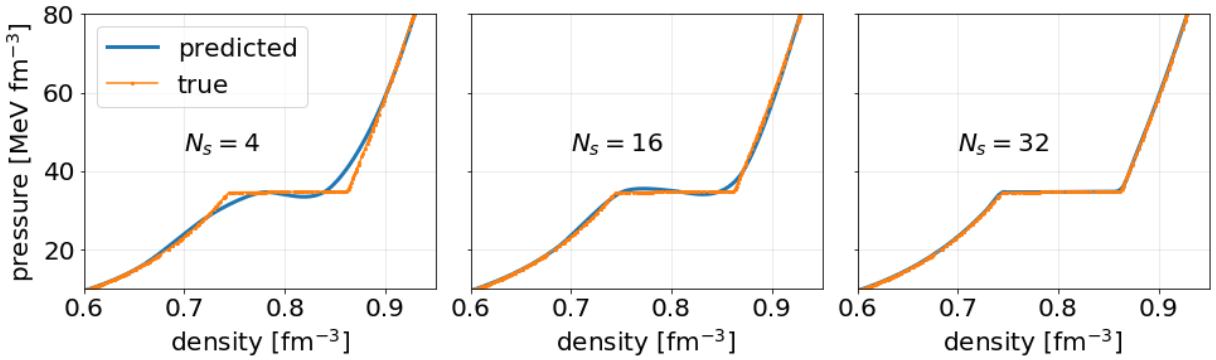

FIG. 10. Quark matter pressure as a function of density for a given proton fraction. The orange curve shows the results from more expensive simulation and the blue curve represents prediction using our methodology. Note as we increase the size of the ensemble of solvers (N_s) directed to find the critical points, the accuracy of the surrogate obtained from a single learning step using a thin-plate RBF improves.

made to 87 tables with 108 parameters. Later, Matteoli and Mansoori[64] suggested a considerably simplified form with only 21 parameters and two forms (small and large separation r), allowing for the possibility of extending the fits to mixtures. Later, Morsali *et al.*[65], reiterate that fits should not have too many parameters, and with improved simulation data, provide a fit with only 11 parameters, again with two functional forms valid at small and large separations.

For strongly coupled plasma studies, the one-component plasma (OCP) is the simplest model used. The OCP is a system of particles with charge Ze interacting through a repulsive Coulomb potential $rV(r) = Z^2e^2$, and a uniform, neutralizing background. Here, Z is the charge number of a given particle, and e is the fundamental charge. The statistical properties of the OCP can be described in terms of a single parameter, the Coulomb coupling parameter $\Gamma = (Z^2e^2)/(aT)$, where $a = (4\pi n/3)^{-1/3}$ is the Wigner-Seitz radius with n being the total ionic number density, and T is the temperature

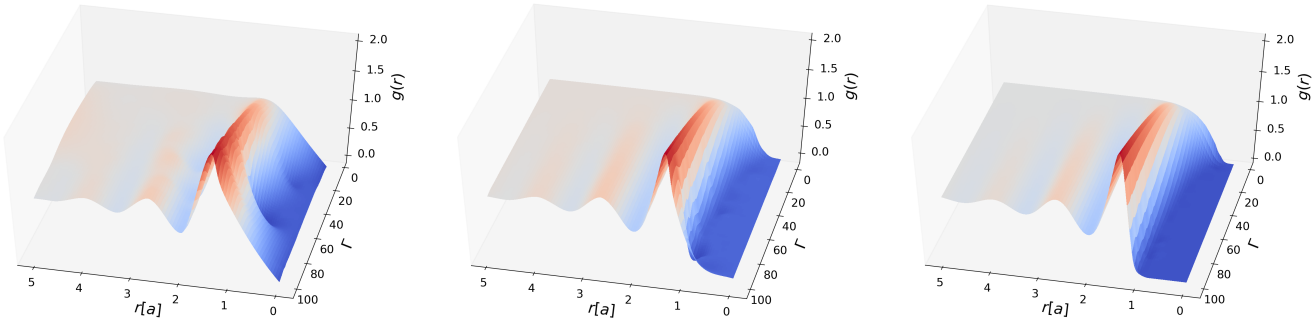

FIG. 11. Illustration of using online learning to refine a surrogate for a radial distribution of OCP obtained from MD using an ensemble of N_s Nelder-Mead optimizers. The optimizers search for critical points, while an estimator learns the surface from the sampled points. Validity of the surrogate is defined as in Eq. (15), with $tol_{max} = 10^{-6}$ and $tol_{sum} = 10^{-3}$, and *train*, *converged*, *data*, and *metric* as defined in Section . The surrogates shown were produced using ensembles of N_s of 4, 8, and 16, respectively. Note that $N_s = 4$ will produce a similar surrogate to $N_s = 16$, either by performing multiple iterations to *test* validity (as described in Section) or by setting *warm* (the required number of model evaluations per iteration) to a large enough number that each of the four optimizers is respawned roughly three times.

in energy units (note that the charge q is represented here in Gaussian-cgs units). Rogers *et al.*[66] employed Monte Carlo data to generate the OCP structure factor in tabular form, which requires a special request (and cost) for distribution. Brettonet and Derouiche[67] provided a fit to the OCP structure factor in terms of only two parameters, one of which was found to be a constant. However, this fit lacked the accuracy of other approaches. Desbiens *et al.*[68] have parametrized the RDF of the OCP using MD data, where their fit model was motivated by the Matteoli and Mansoori form[64]. To achieve high accuracy, it was necessary to fit weak and strong coupling functional forms separately, with five and nine parameters, respectively. Because there is a Fisher-Widom (FW) line[69–72], which delineates the transition to oscillatory behavior in the RDF, different functional forms are needed. These examples illustrate the challenges with tabulating or fitting data to high accuracy, and because these challenges are for a model system with a single parameter Γ, it is not clear that they generalize to more complex systems such as binary ionic mixtures (BIM), ternary ionic mixtures (TIM) or Lennard-Jones mixtures where we have additional parameter space.

RDFs were computed with MD for three systems to explore both the type (*i.e.*, range and repulsive/attractive) of interaction and the role of dimensionality. In all cases, the MD code LAMMPS[73] was used to produce the data with standard techniques. The equations of motion were first integrated in the canonical ensemble, with constant particle number, volume, and temperature maintained using a Nosé-Hoover thermostat, over 10^5 timesteps, to establish thermodynamic equilibrium at the desired temperature. Then, the production runs were carried out in the micro-canonical ensemble with $t = 1000\,\omega_p^{-1}$, where ω_p^{-1} is a characteristic oscillation period for the system. To help improve the quality of our data, we reduced statistical fluctuations through the use of large particle numbers ($N = 2048$) and long runs. The RDFs $g(r;\mathbf{x})$ were calculated with 1024 bins in the range $0 < r < L/2$, where L is the simulation system size, and $\mathbf{x}$ are the input parameters of system. Note that for binary mixtures, we have three different RDFs $g_{ij}(r;x)$ where i and j are the species indexes.

In each of the three cases below, all LAMMPS model evaluations are archived to a model evaluation database, as in Method A. We used "lattice" sampling with an ensemble of 40 Nelder-Mead solvers at the default configuration to generate the model evaluations, and interpolation with a thin-plate RBF to generate the surrogate. We defined *test* validity as in Eq. (15) with $tol_{max} = 10^{-6}$ and $tol_{sum} = 10^{-3}$, and *train*, *converged*, *data*, and *metric* as defined in Section . Learned surrogates were saved to a surrogate database as in Method B.

One-Component Plasma. The one-component plasma (OCP) is a model that assumes a system of point ions with charge q at a temperature T embedded in a uniform, neutralizing background. The thermodynamic state of the OCP system is entirely determined by the coupling parameter Γ, so each MD simulation computes the RDF with the form $g(r,\Gamma)$. Since the RDF enters into the calculations of many quantities in many forms, we also choose to calculate the related Salpeter screening potential, which is defined in terms of the RDF as:

$$\beta H(r) = \frac{\Gamma}{r} + \log\big[g(r)\big], \tag{16}$$

where $\beta = 1/T$, and r is in units of the Wigner-Seitz radius a. The screening potential $H(r)$ is used to estimate the enhancement factor of nuclear reaction rates in dense plasmas.

For the OCP, the radial coordinate r was defined to be in the range $[0, L/2]$, where L is the simulation box size, and the coupling parameter was taken be in the range of $[0.1, 100]$, which spans both weakly and strongly coupled regimes. Figure 11 illustrates the sampling process of the framework with the surface of the OCP RDF $g(r,\Gamma)$, mapped using an ensemble of local optimizers. The surrogate surface is interpolated on a uniform grid,

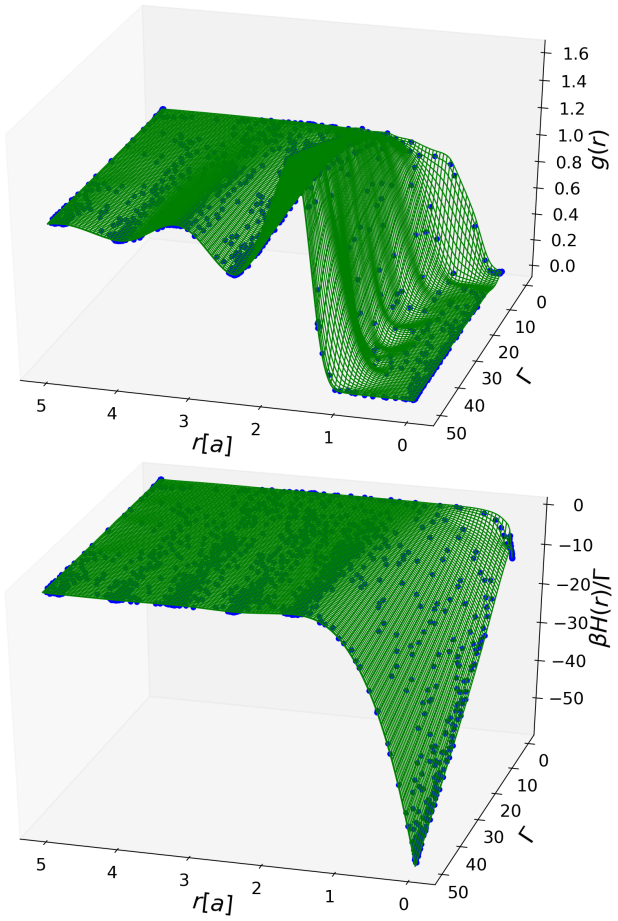

FIG. 12. MD data (blue dots) compared to the surrogate (green surface) for the OCP system in (r, Γ) space, for (a) RDF $g(r)$ (b) screening potential $H(r)$. The initial search domain was defined as $\Gamma \in [0.1, 50]$ and $r[a] \in [0, 5]$, covering both the weakly and strongly coupled regimes. We used a lattice sampler with an ensemble of 40 Nelder-Mead solvers at the default configuration, a definition of *test* validity as in Eq. (15) with $tol_{max} = 10^{-6}$ and $tol_{sum} = 10^{-3}$, and *train*, *converged*, *data*, and *metric* as defined in Section . The resulting surrogate for $g(r)$, $H(r)$, and the associated model evaluations, are stored to disk (see Method E) for programmatic access by coarse-grained codes.

and as we increase the sampling, the surface resolution improves considerably, and the surrogate captures details very accurately across physical regimes. In Figure 12, we show the predicted RDF and screening potential in (Γ, r) space. The blue dots are directly calculated with MD, while the green lines are calculated with the learned surrogates. We observe excellent agreement between the results produced with the surrogates, and MD, across the defined parameter space.

Next, we examined the predictive accuracy of surrogates learned with the procedure above, with regard to thermodynamic quantities, such as the internal energy and pressure. For an OCP, the internal energy and pressure are given by:

$$\frac{\beta U}{N} = \frac{3}{2}\Gamma \int_0^\infty r\big[g(r) - 1\big]dr, \tag{17}$$

$$\beta\frac{P}{n} = 1 + \frac{\Gamma}{2}\int_0^\infty r\big[g(r) - 1\big]\,dr, \tag{18}$$

where N is the number of particles, and n the density. We trained surrogates for the OCP as above, and repeated the procedure for several different combinations of N_s and tol_{sum}. Note that surrogates were not trained on energy and pressure, but were trained and scored as in Eq. (15). We generated surrogates using all possible combinations of $N_s = \{4, 10\}$ and $tol_{aum} = \{10^{-4}, 10^{-3}\}$. We then used Eqs. (17)-(18) to calculate the pressure and energy using our surrogates, and then compared to similar results produced with MD. We found that the surrogates show an increase in predictive accuracy when $N_s = 10$, as can be seen in Figure 13. Note that strong oscillations are observed in the surrogate's predictions of both the pressure and energy when $N_s = 4$, and that the number of peaks and their amplitude are more substantial at high values of Γ. Indeed, with $N_s = 4$, the surrogates struggle for predictive accuracy in the region of high coupling. When we increase to N_s to 10, we have a higher volume of training data near the critical points, and thus surrogate quality improves around the critical points. In this case, the predicted pressure and internal energy agree very well with MD, where the relative deviation from ground truth (MD) is less than 10^{-4}. That this trend holds across the full range of coupling parameter examined.

Lennard-Jones fluid. We now consider a LJ liquid in 2 spatial dimensions, where the system is composed of N particles of mass m randomly distributed in the simulation box and interacting through the LJ potential given by:

$$V(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right], \tag{19}$$

with depth of the potential, ϵ, and σ the relative distance at which the interaction between pairs of two particles is zero. The LJ potential is known to give a reasonable description of interactions of atoms in rare-gas systems, which is the limit to which the attractive component is derived. For the LJ potential, we used the reduced units given by: $T^* = T/\epsilon$, $r^* = r/\sigma$ and $\rho^* = \rho\sigma^3$, where ρ^* is the dimensionless density. The resulting RDF for the LJ system then has the form: $g(r^*, T^*, \rho^*)$.

We trained surrogates for the LJ system on $g(r^*, T^*, \rho^*)$, as opposed to $g(r, \Gamma)$ in the previous section for the OCP. We compared our results to calculations performed with MD for different values of T^* and ρ^*. Figure 14 compares the LJ RDF generated with MD with that of the surrogate, for three different values of temperature and density. As can be seen from the figure, the surrogates very accurately predict the results from MD across the entire range, and for different T^* and ρ^*.

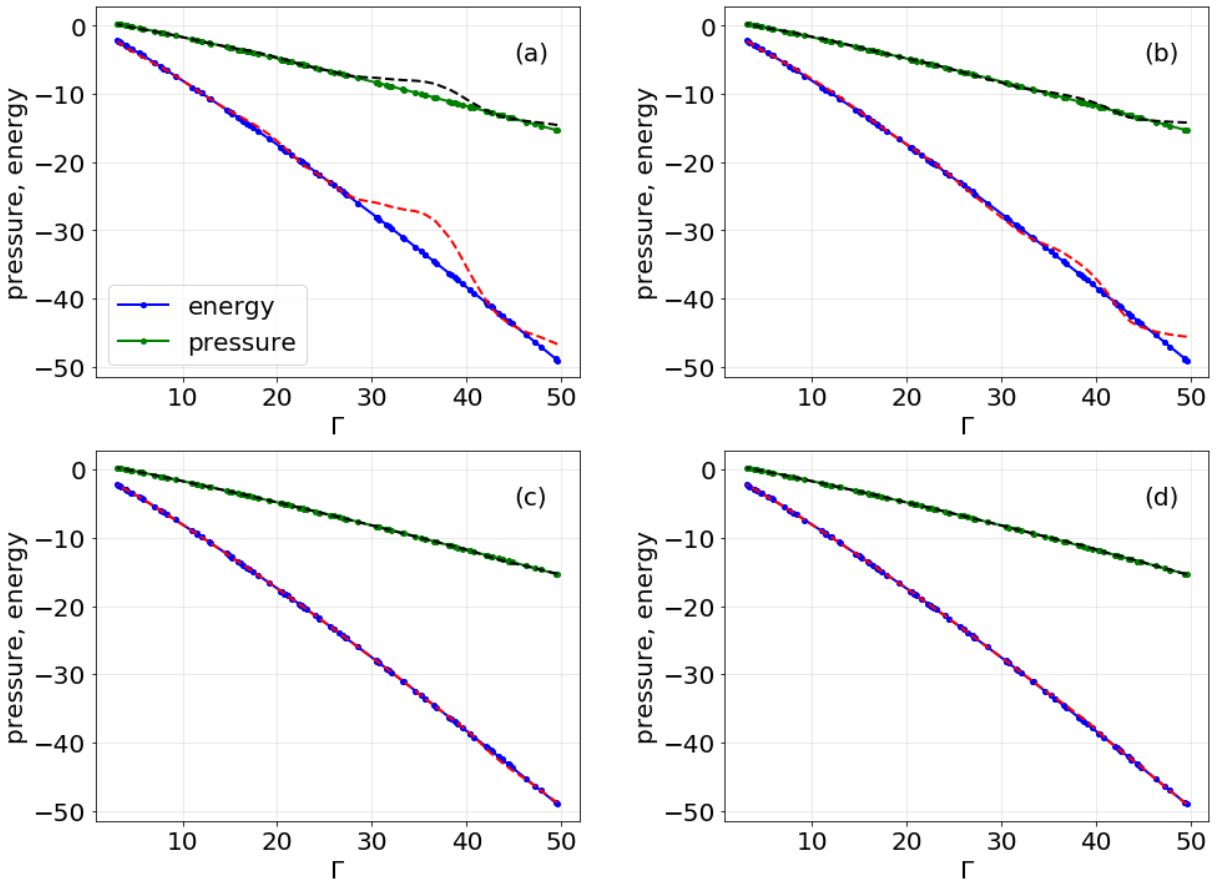

FIG. 13. Energy ($\beta U/N$) and Pressure ($\beta P/\rho$) predicted by a learned surrogate (dashed lines) for an OCP, as compared to results of MD simulations (dotted lines). Surrogates were trained using thin-plate RBF interpolation, with lattice sampling directed by an ensemble of N_s Nelder-Mead optimizers. Note that surrogates were not trained on energy and pressure, but used a definition of *test* validity as in Eq. (15), with *data* and *metric* as defined in Section , and $tol_{max} = 10^{-6}$. We define *train* and *converged* identically to *test*, and (a) $N_s = 4$, $tol_{sum} = 10^{-3}$; (b) $N_s = 4$, $tol_{sum} = 10^{-4}$; (c) $N_s = 10$, $tol_{sum} = 10^{-3}$; (d) $N_s = 10$, $tol_{sum} = 10^{-4}$. When comparing energy and pressure calculated by MD with that produced by the surrogates, the surrogates showed an increase in predictive accuracy when $N_s = 10$, as opposed to 4. This improvement can be attributed to the higher volume of training data near the critical points.

Binary Ionic Mixture. Finally, we consider a BIM model, which is an extension of the OCP allowing for two separate species embedded in a neutralizing electronic background. The ion-ion interaction potentials are modeled with the Coulomb potential

$$V_{ij}(r) = \frac{Z_i Z_j e^2}{r}, \tag{20}$$

where the Z_i is the charge number of the i^{th} species. As with the OCP simulations, the long-range Coulomb forces are handled through an Ewald summation algorithm. For the binary mixture, we introduce the more general coupling parameter

$$\Gamma_{ij} = \frac{Z_i Z_j e^2}{aT}. \tag{21}$$

Letting $\Gamma = e^2/(aT)$ be the proton-proton coupling parameter, the RDFs of the BIM model will subsequently have the form: $g_{ij}(r; \Gamma, c, Z_i, Z_j)$, where $c = n_1/n$ is the concentration of one of the species.

Figure 15 plots the RDFs for three different mixtures of chemical elements present in white dwarfs[2,74,75], Al-Fe, Na-Mg, C-O, as produced with MD. We also generated surrogates as in the previous sections, and compared the results to the MD data. We observed excellent agreement between the surrogates and MD, despite the higher dimensions of the BIM RDF parameter space.

DISCUSSIONS

We presented an online learning approach designed to produce valid surrogates for a chosen quality metric. While the approach works well in generating valid surrogates for existing data, it can also be applied to generating valid surrogates with regard to future data. We demonstrated an application of online learning, where the

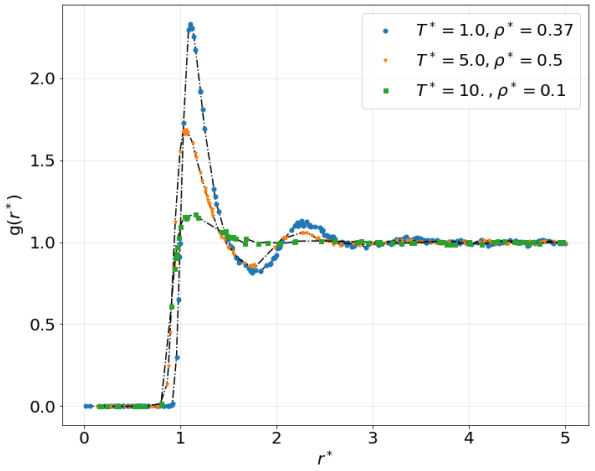

FIG. 14. RDF $g(r)$ for a 2-D Lennard-Jones (LJ) liquid. The markers (squares, etc) indicate data generated from MD. The lines show the surrogate-predicted results, with the surrogates trained. The physical quantities are in LJ units. For the one-fluid LJ system, we observed the same high level of predictive accuracy, with regard to various thermodynamic state points, as we found for the surrogates for OCP and BIM.

selection of training data is performed using a sampling strategy, and an iterative approach is used to improve surrogate validity versus the chosen metric. We gave evidence for the conjecture that if the critical points of the model's response surface are known, a robust estimator (e.g. thin-plate RBF interpolation or a MLP neural network) should be able to produce a surrogate that exactly reproduces the behavior of some more expensive model. We presented an "optimizer-directed" sampling strategy that was shown to be effective at sampling the critical points of a model's response surface. We then compared the efficiency of different sampling strategies in learning surrogates that are valid for benchmark functions, even in the presence of newly sampled data. Note that if for whatever reason, the surrogate was found to be invalid about newly acquired data, our online approach can be used to improve the surrogate iteratively.

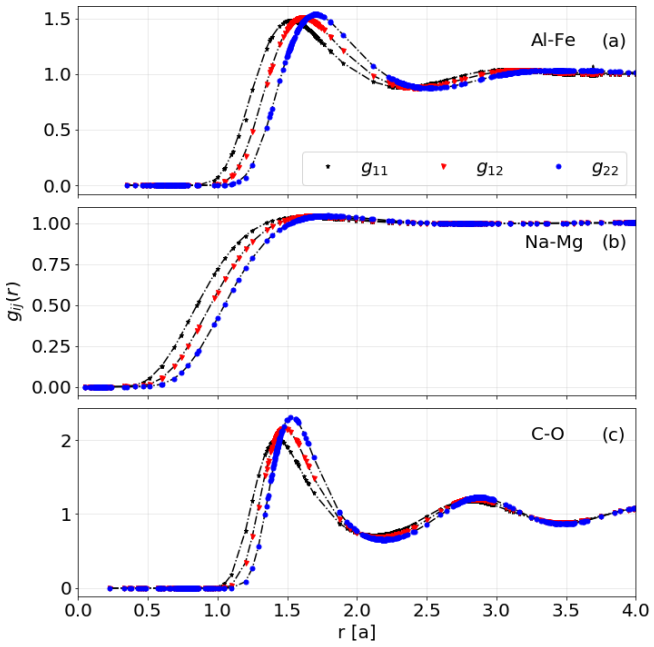

FIG. 15. RDFs $g_{ij}(r)$ of various binary ionic mixtures: Na-Mg, Al-Fe and C-O. The markers (dots) correspond to data generated by MD. The lines show the surrogate-predicted results. In each case: (a) $\Gamma = 0.1, c = 0.8, Z_1 = 11, Z_2 = 12$; (b) $\Gamma = 1, c = 0.6, Z_1 = 13, Z_2 = 2$; (c) $\Gamma = 2.5, c = 0.5, Z_1 = 6, Z_2 = 8$, the predicted results accurately reproduce the results from MD.

For selected benchmark functions, we used a 'sparse' sampling approach that produced new draws at the least-populated points in parameter space. We compared a more "traditional" sparse sampling that finds and samples at the least-populated points each draw, to an "optimizer-directed" approach where each initial draw is used as starting points for an optimizer that runs to termination. We found that, at first blush, it would seem that the traditional sampling strategy outperforms the optimizer-directed strategy for all benchmark functions (see Table I). It should be noted, however, that the metric we used was based on the error in the surrogate's predicted value versus truth, averaged out over the entire response surface. Hence, it should not be surprising that our methodology produced surrogates that are, on average, of high quality across the entire parameter range. One, therefore, might conclude that a traditional sampling strategy, especially one that provides more diffuse sampling than an optimizer-directed strategy, is more efficient at generating valid surrogates when the average misfit across parameter space measures the validity. However, we note that the default optimizer configuration was used in testing the efficiency of the sampling strategy, and *tuning* the optimizer may make a substantial difference in the efficiency of the optimizer-directed strategy. We tested two different optimizers and noted their convergence behavior using the default settings. We found that the choice of optimizer can affect the efficiency by over an order of magnitude (see Table II), and that the use of stronger termination conditions can also impact the efficiency by order of magnitude (see Figure 7). In that light, given that one does some upfront work to tune the optimizer for the given problem, the optimizer-directed approach can *easily* be more efficient than a traditional sampling approach.

Importantly, we also noted that the metric we used made no guarantee of the quality of the surrogate *in the neighborhood of the critical points*. Returning to our conjecture, the critical points of the response surface are the key points to find to guarantee the long-term validity of the surrogate as new data is collected. We found that an optimizer-directed approach is much better at minimizing the model error in the neighborhood of the critical points (see Figure 3), even when the metric does not call

for that explicitly. Conversely, blind to the response surface, a traditional sampling strategy generally demonstrated a much larger misfit near the critical points. Thus, using a metric that judges the quality of the surrogate by the misfit at the critical points should produce high-quality surrogates with an optimizer-directed approach with even greater efficiency.

The critical points of the response surface are where the exciting physics occurs, thus providing additional motivation to ensure the misfit near the critical points is minimal. To demonstrate this, we applied our methodology to two test problems. We showed that we could efficiently learn surrogates for equation-of-state calculations of dense nuclear matter, yielding excellent agreement between the surrogate and model both across the parameter range and specifically in the region that includes physically relevant features, such as a phase transition. We also showed our methodology can produce highly-accurate surrogates for radial distribution functions for expensive molecular dynamics simulations for neutral and charged systems of several dimensions across an extensive range of thermodynamic conditions. Note also that while our demonstrations focused on fluid dynamics, the methodology and associated code are agnostic to the domain science and can be utilized for a wide variety of problems.

A standard metric used to determine the validity of a surrogate is the model error, as in Eq. (3). This definition assesses the quality of the surrogate by measuring the distance of the surrogate from the observed data. Unfortunately, if one has only a small set of observed data that is not representative, then any learned surrogate will very likely become invalidated with the addition of new data. A potentially more robust assessment of model quality considers training a surrogate with a statistical metric, such as the *expected* model error. The expected model error can be defined to take into account any knowledge about the data-generating distributions (for input and output values) and any uncertainty in the input and output parameters of the model. Complex real-world models are often non-deterministic; thus, an appropriate goal is to either find a surrogate that is guaranteed to be accurate under uncertainty or a surrogate that is guaranteed to be robust under uncertainty. With some minor adjustments, such as adding a strategy to timestamp or invalidating training data, our methodology can be leveraged to build and maintain accurate surrogates for time-dependent models. In future work, we will apply our methodology to produce surrogates that are guaranteed to be either accurate or robust under uncertainty and similarly demonstrate the ability to guarantee the accuracy of surrogates for time-dependent models.

The code implemented for our methodology facilitates saving the learned surrogates to a database, where they can be easily utilized within coarse-grain calculations and codes, as in[19]. Similarly, any sampled data used in this work is seamlessly saved to a database. The code, as well as the sampled data and learned surrogates, relevant to this work are under review for public release.

METHODS: IMPLEMENTATION

We present here some of the components of the method and implementation to *mystic*.

A. Cached model evaluations

Given an expensive `model`, we augment the model by linking the model to a database of model evaluations:

```
model = mystic.cache.cached('model')(model)
```

where the resulting database is named `'model'`, and the model is now augmented by with programmatic access to the database:

```
cache = model.__cache__()
```

and an inverted model

```
inverted_model = model.__inverse__
```

which also hooks into the database, and can be used in solving for maxima. Alternately, we can augment the model with:

```
model = WrapModel('model', model, cached=True, rnd=False)
```

which additionally augments the model with sampling and distance methods, as detailed in B and D, respectively. We can also access the database of model evaluations directly from disk

```
cache = mystic.cache.archive.read('model')
```

and, regardless of how the cache was accessed, the data can be loaded into a `mystic.math.legacy.dataset` with:

```
data = as_dataset(cache)
```

or a `mystic.monitor` with:

```
monitor = as_monitor(cache)
```

B. Sampling model evaluations

The samplers available in `mystic.samplers` provide an interface to optimizer-directed and more traditional sampling methods. Here we combine a `SparsitySampler` with a `NelderMeadSimplexSolver` to produce an ensemble of Nelder-Mead optimizers that start where the database is the most sparse, and run to termination:

```
kwds = {solver:NelderMeadSimplexSolver, dist:None}
sampler = SparsitySampler(bounds, model, pts, **kwds)
sampler.sample_until(terminated=all)
```

where, `bounds` indicates the lower and upper bounds for each input parameter, `dist` is a distribution object or similar random number generator that provides additional noise in the sampling, and `pts` is the integer number of optimizers to use *in each direction*. Hence, `pts = 4` will spawn 4 optimizers to search for minima, and 4 optimizers to search for maxima. Here, `model` is an expensive model as generated in A. For `solver`, we are using a Nelder-Mead solver at the default configuration (see F), as opposed to passing a configured solver instance. A sampler also provides traditional (not optimizer-directed) sampling, with:

```
sampler.sample()
```

so, a `SparsitySampler` with `pts = 4` will sample 8 data points in total, located at the 8 most sparse coordinates in the database. Alternately, if a `WrapModel` is used,

```
model.sample(bounds, -pts, sampler=SparsitySampler, **kwds)
```

then setting `pts = 4` will spawn 4 minima-seeking and 4 maxima-seeking optimizers, while

```
model.sample(bounds, pts, sampler=SparsitySampler, **kwds)
```

will use traditional sampling to sample 8 total points.

C. Learning a surrogate

A surrogate is generated with a learning strategy, based on an estimation method such as interpolation or machine learning. We learn a surrogate with interpolation by using `InterpModel`, essentially a `WrapModel` which uses `mystic.math.interpolate.interpf` to produce an interpolated surrogate. For example:

```
kwds = {method:'thin_plate', noise:1e-8, smooth:0}
surrogate = InterpModel('surrogate', model, **kwds)
```

generates a `surrogate` for `model` that attempts to best satisfy Eq. (5) using a 'thin-plate' RBF. Here, `smooth:0` forces the interpolated function to go through the nodal points, and `noise:1e-8` adds Gaussian noise with an amplitude of `1e-8` to remove duplicate inputs (and thus avoid a singular matrix. Thus once `surrogate.fit()` is called, or the surrogate is evaluated, `surrogate(x)` closely approximates `model(x)` for all data in the `'model'` database. Whenever `model` is evaluated, and thus more data is added to the model database, we can call `surrogate.fit()` to trigger an update to the interpolated surrogate. We can save the surrogate to disk for later use with:

```
mystic.cache.function.write(surrogate, 'surrogate.db')
```

Alternately, we can use machine learning to generate a surrogate. For example, we use a `MLPRegressor` and a `StandardScaler` from *scikit-learn* and `MLData`, `Estimator`, and `LearnedModel` from *mystic* generate a surrogate from a neural network (NN). We generate a training set that includes all the data, as we will be testing on yet-to-be-acquired data:

```
x, y = data.coords, data.values
mld = MLData(x, x, y, y)
```

as indicated in Fig. 1. We next build an estimator, and as training a NN is non-deterministic, we use `improve_score` from *mystic* to iteratively improve the best-fit estimator:

```
args = {hidden_layer_sizes:(100,75,50), max_iter:1000,
        n_iter_no_change:5, solver:'lbfgs'}
est = Estimator(MLPRegressor(**args), StandardScaler())
est = improve_score(est, mld, tries=10, verbose=True)
```

Here, `args` are configuration hyperparameters for the `MLPRegressor`. `StandardScaler` will use the default configuration. The estimator `est` is considered best-fit to the data when `tries=10` successive iterations fail to yield improvement. Similarly to how an `InterpModel` uses interpolation to learn a surrogate for an expesive model, a `LearnedModel` uses the above-configured estimator to produce a surrogate from a NN

```
kwds = {estimator:est.estimator, transform:est.transform}
surrogate = LearnedModel('surrogate', model, **kwds)
```

D. Validity metrics

Surrogate validity is defined in terms of a metric, which measures the distance between the surrogate and the model (or model evaluations). One of the most common metrics is the square of the pointwise difference between the outputs:

```
dist = [(surrogate(x) - model(x))**2 for x in data.coords]
```

and this is indeed the default metric in an `ErrorModel`

```
error = ErrorModel('error', model, surrogate, metric=None)
dist = [error(x) for x in data.coords]
```

where, alternate metrics can be provided of the form `metric(y, y')`. A less common but more robust metric is the graphical distance, which is essentially the *shortest normal line* from the model evaluated at `x` to the surrogate at the point of tangency. The graphical distance is available at `mystic.math.distance.graphical_distance`

```
dist = graphical_distance(surrogate, data, hausdorff=True)
```

or directly from a surrogate built as in Method C

```
dist = surrogate.distance(data)
```

where `hausdorff` indicates whether or not we include Δ_x. Once a metric has been defined, we can find the surrogate that is optimally valid, in terms of the surrogate hyperparameters, by using an optimizer to minimize `dist`. Alternately, we can define a validity threshold, such as

```
valid = max(dist) <= 1e-4 and mean(dist) <= 1e-5
```

that validates the surrogate when `valid = True`.

E. Using stored surrogates

Learned surrogates are stored to disk (as in Method C), and can be accessed programmatically. When the surrogates are evaluated for given inputs, they provide a reasonable approximation for results generated by the more expensive model. For example, we load the surrogate for the binary mixture from disk with:

```
g = mystic.cache.function.read("BIM.db")
```

and then evaluate the surrogate:

```
g11, g12, g22 = g(r, Gamma, c, Z1, Z2)
```

where `r` is the radial coordinate in Wigner-Seitz radius, `Gamma` is the proton-proton coupling parameter, `c` is the concentration of species 1 with nuclear charge `Z1`, and `Z2` is the nuclear charge of species 2.

F. Default optimizer configuration

The defaults for the `NelderMeadSimplexSolver` are:

```
defaults = {maxiter: None, maxfun: None, xtol: 1e-4,
            ftol: 1e-4, radius: 0.05, adaptive: False}
```

where `maxiter` is the maximum number of iterations, `maxfun` is the maximum number of function evaluations, `xtol` is the acceptable absolute change in each parameter for convergence, `ftol` is the acceptable absolute change in the objective for convergence, `radius` is the percent change for initial simplex values, and `adaptive` is true if adaptive parameters should be used. The Nelder-Mead optimizer uses a candidate relative tolerance (CRT) termination, specifically `CRT(xtol, ftol)`[24].

The defaults for the `PowellDirectionalSolver` are:

```
defaults = {maxiter: None, maxfun: None, ftol: 1e-4,
            gtol: 2, imax: 500, xtol: 1e-4, direc: None}
```

where `maxiter`, `maxfun`, and `ftol` are defined as above, `gtol` is the maximum iterations to run without improvement, `imax` is the line-search maximum iterations, `xtol` is the line-search error tolerance, and `direc` is the the initial direction set (which, if not provided, will use the identity matrix). Powell's optimizer uses a normalized change over generation (NCG) termination, specifically `NCG(ftol, gtol)`[24].

Thus, the Nelder-Mead solver will stop when the absolute difference in both $\mathbf{x}$ and $f(\mathbf{x})$ over one iteration is less than 10^{-4}, while Powell's solver will stop when the normalized absolute difference of $f(\mathbf{x})$ over `gtol` iterations is less than 10^{-4}.

G. Hartmann's function coefficients

The coefficients of the 6-D Hartmann's function Eq. (12) are given by

$$\alpha = \begin{vmatrix} 1.0 \\ 1.2 \\ 3.0 \\ 3.2 \end{vmatrix}, A = \begin{vmatrix} 10 & 3 & 17 & 3.50 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{vmatrix}$$

and

$$P = 10^{-4} \begin{vmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{vmatrix}$$

while the coefficients for the 3-D Hartmann's function are:

$$A = \begin{vmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{vmatrix} \text{ and } P = 10^{-4} \begin{vmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{vmatrix}.$$

H. Equations of State

The energy per baryon of nuclear matter composed of neutrons and protons with densities ρ_n and ρ_p are given, respectively, by:

$$\begin{aligned} \frac{E}{A}(y_p, \rho) = {} & \frac{3}{5}\left(\frac{\hbar^2}{2m}\rho\right)^{\frac{2}{3}} F_{5/3} \\ & + \frac{1}{8} t_0 \rho \left[2(x_0+2) - (2x_0+1)F_2\right] \\ & + \frac{1}{48} t_3 \rho^{\alpha+1} \left[2(x_3+2) - (2x_3+1)F_2\right] \\ & + \frac{3}{40}\left(\frac{3\pi^2}{2}\right)^{\frac{2}{3}} \rho^{\frac{5}{3}} \Big[\left[t_1(x_1+2) + t_2(x_2+1)\right] F_{\frac{5}{3}} \\ & + \frac{1}{2}\left[t_2(2x_2+1) - t_1(2x_1+1)\right] F_{\frac{8}{3}}\Big], \end{aligned} \tag{22}$$

$$F_m(y_p) = 2^{m-1}\left[y_p^m + (1-y_p)^m\right], \tag{23}$$

$$\rho = \rho_n + \rho_p, \quad y_p = \rho_p/\rho. \tag{24}$$

The parameters $x_{1\ldots3}$, $t_{1\ldots3}$ and α are fitted to reproduce properties of nuclei, such as binding energy, or of neutron stars, such as the neutron star radii.

CODE AVAILABILITY

The code, as well as the sampled data and learned surrogates, relevant to this work are under review for public release.

ACKNOWLEDGMENTS

Research presented in this article was supported by Los Alamos National Laboratory under the Laboratory Directed Research and Development program (project numbers 20190005DR, 20200410DI, and 20210116DR), by the Department of Energy Advanced Simulation and Computing under the Beyond Moore's Law Program, and by the Uncertainty Quantification Foundation under the Statistical Learning program. Los Alamos National Laboratory is operated by Triad National Security, LLC, for the National Nuclear Security Administration of U.S. Department of Energy (Contract No. 89233218CNA000001). The Uncertainty Quantification Foundation is a nonprofit dedicated to the advancement of predictive science through research, education, and the development and dissemination of advanced technologies. The authors would like to thank Jeff Haack for his very useful feedback on the manuscript. This document is LA-UR-20-24947.

AUTHOR CONTRIBUTIONS STATEMENT

Diaw, McKerns, and Murrillo conceived the project. McKerns wrote the algorithm and implemented it into mystic. Diaw and McKerns prepared figures. Sagert provided the EOS code. Numerical data were analysed by all authors. All authors reviewed the manuscript.

ADDITIONAL INFORMATION

Competing financial interests: The authors declared no competing financial interests.

[1] P. V. Coveney, J. P. Boon, and S. Succi, Bridging the gaps at the physics-chemistry-biology interface, Philosophical Transactions of the Royal Society of London Series A **374**, 20160335 (2016).

[2] B. Paxton, J. Schwab, E. B. Bauer, L. Bildsten, S. Blinnikov, P. Duffell, R. Farmer, J. A. Goldberg, P. Marchant, E. Sorokina, A. Thoul, R. H. D. Townsend, and F. X. Timmes, Modules for Experiments in Stellar Astrophysics (MESA): Convective Boundaries, Element Diffusion, and Massive Star Explosions, ApJS **234**, 34 (2018), arXiv:1710.08424 [astro-ph.SR].

[3] K. Kadau, T. C. Germann, P. S. Lomdahl, and B. L. Holian, Microscopic View of Structural Phase Transitions Induced by Shock Waves, Science **296**, 1681 (2002).

[4] C. L. Fryer, A. Diaw, C. J. Fontes, A. L. Hungerford, J. Kline, H. Johns, N. E. Lanier, S. Wood, and T. Urbatsch, Designing radiation transport tests: Simulation-driven uncertainty-quantification of the COAX temperature diagnostic, High Energy Density Physics **35**, 100738 (2020), arXiv:1912.10534 [physics.app-ph].

[5] C. Ticknor, J. D. Kress, L. A. Collins, J. Clérouin, P. Arnault, and A. Decoster, Transport properties of an asymmetric mixture in the dense plasma regime, Phys. Rev. E **93**, 063208 (2016).

[6] L. G. Stanton, J. N. Glosli, and M. S. Murillo, Multiscale molecular dynamics model for heterogeneous charged systems, Phys. Rev. X **8**, 021044 (2018).

[7] L. A. Collins, J. D. Kress, S. F. Mazevet, and M. P. Desjarlais, Quantum Molecular Dynamics Simulations of Dense Plasmas, IEEE Transactions on Plasma Science **33**, 586 (2005).

[8] J. D. Kress, J. S. Cohen, D. A. Horner, F. Lambert, and L. A. Collins, Viscosity and mutual diffusion of deuterium-tritium mixtures in the warm-dense-matter regime, Phys. Rev. E **82**, 036404 (2010).

[9] E. W. Brown, B. K. Clark, J. L. DuBois, and D. M. Ceperley, Path-integral monte carlo simulation of the warm dense homogeneous electron gas, Phys. Rev. Lett. **110**, 146405 (2013).

[10] T. Dornheim, J. Vorberger, S. Groth, N. Hoffmann, Z. A. Moldabekov, and M. Bonitz, The static local field correction of the warm dense electron gas: An ab initio path integral Monte Carlo study and machine learning representation, J. Chem. Phys. **151**, 194104 (2019), arXiv:1907.08473 [physics.plasm-ph].

[11] J. Schmidt, M. Marques, S. Botti, and M. Marques, Recent advances and applications of machine learning in solid-state materials science, npj Computational Materials **5**, 10.1038/s41524-019-0221-0 (2019).

[12] Y. Liu, T. Zhao, W. Ju, and S. Shi, Materials discovery and design using machine learning, Journal of Materiomics **3**, 10.1016/j.jmat.2017.08.002 (2017).

[13] V. Barros, C. Field, D. Dokken, M. Mastrandrea, K. Mach, T. Bilir, M. Chaterjee, K. Ebi, Y. Estrada, R. Genova, B. Girma, E. Kissel, A. Levy, S. MacCracken, P. Mastrandrea, and L. White, eds., *Climate Change 2014 Impacts, Adaptation, and Vulnerability* (Cambridge University Press, New York, 2014).

[14] P. Wigley, P. Everitt, A. van den Hengel, J. Bastian, M. Sooriyabandara, G. McDonald, K. Hardman, C. Quinlivan, P. Manju, C. Kuhn, I. Peterson, A. Luiten, J. Hope, N. Robins, and M. Hush, Fast machine-learning online optimization of ultra-cold-atom experiments, Scientific Reports **6**, 10.1038/srep25890 (2016).

[15] A. Scheinker and S. Gessner, Adaptive method for electron bunch profile prediction, Physical Review Accelerators and Beams **18** (2015).

[16] M. Noack, K. Yager, M. Fukuto, G. Doerk, L. Ruipeng, and J. Sethian, A kriging-based approach to autonomous experimentation with applications to x-ray scattering, Scientific Reports **9**, 10.1038/s41598-019-48114-3 (2019).

[17] N. Lubbers, A. Agarwal, Y. Chen, S. Son, M. Mehana, Q. Kang, S. Karra, C. Junghans, T. C. Germann, and H. S. Viswanathan, Modeling and scale-bridging using machine learning: nanoconfinement effects in porous media, Scientific Reports **10**, 13312 (2020).

[18] A. Diaw, K. Barros, J. Haack, C. Junghans, B. Keenan, Y. W. Li, D. Livescu, N. Lubbers, M. McKerns, R. S. Pavel, D. Rosenberger, I. Sagert, and T. C. Germann,

Multiscale simulation of plasma flows using active learning, Phys. Rev. E **102**, 023310 (2020).
[19] D. Roehm, R. S. Pavel, K. Barros, B. Rouet-Leduc, A. L. McPherson, T. C. Germann, and C. Junghans, Distributed Database Kriging for Adaptive Sampling (D^2 KAS), Computer Physics Communications **192**, 138 (2015).
[20] J.-L. Coulomb, A. Kobetski, M. Caldora Costa, Y. Maréchal, and U. Jonsson, Comparison of radial basis function approximation techniques, COMPEL-The international journal for computation and mathematics in electrical and electronic engineering **22**, 616 (2003).
[21] J. Park and I. W. Sandberg, Universal approximation using radial-basis-function networks, Neural computation **3**, 246 (1991).
[22] Y. Wu, H. Wang, B. Zhang, and K.-L. Du, Using radial basis function networks for function approximation and classification, ISRN Applied Mathematics **2012** (2012).
[23] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (MIT press Cambridge, MA, 1986) pp. 318–362.
[24] M. McKerns, P. Hung, and M. Aivazis, mystic: highly-constrained non-convex optimization and UQ (2009), http://pypi.python.org/pypi/mystic.
[25] M. McKerns, L. Strand, T. J. Sullivan, A. Fang, and M. Aivazis, Building a framework for predictive science, in *Proceedings of the 10th Python in Science Conference* (2011) pp. 67–78, http://arxiv.org/pdf/1202.1056.
[26] F. Khoshnoud, I. I. Esat, C. W. de Silva, M. M. McKerns, and H. Owhadi, Self-Powered Dynamic Systems in the Framework of Optimal Uncertainty Quantification, Journal of Dynamic Systems, Measurement, and Control **139**, 10.1115/1.4036367 (2017).
[27] H. Owhadi, C. Scovel, T. J. Sullivan, M. McKerns, and M. Ortiz, Optimal Uncertainty Quantification, SIAM Rev. **55**, 271 (2013).
[28] T. J. Sullivan, M. McKerns, D. Meyer, F. Theil, H. Owhadi, and M. Ortiz, Optimal uncertainty quantification for legacy data observations of Lipschitz functions, ESAIM Math. Model. Numer. Anal. **47**, 1657 (2013).
[29] P.-H. T. Kamga, B. Li, M. McKerns, L. H. Nguyen, M. Ortiz, H. Owhadi, and T. J. Sullivan, Optimal uncertainty quantification with model uncertainty and legacy data, Journal of the Mechanics and Physics of Solids **72**, 1 (2014).
[30] C. W. Li, M. M. McKerns, and B. Fultz, A raman spectrometry study of phonon anharmonicity of zirconia at elevated temperatures, Journal of the American Ceramic Society **94**, 224 (2011).
[31] J. Belak, D. Orikowski, S. Applegate, H. Owhadi, and M. McKerns, Quantifying model uncertainty (2012), lLNL-PRES-585774.
[32] M. McKerns, F. Alexander, K. Hickmann, T. Sullivan, and D. Vaughn, Optimal Bounds on Nonlinear Partial Differential Equations in Model Certification, Validation, and Experiment Design (2020), https://arxiv.org/abs/2009.06626.
[33] M. McKerns, L. Roth, N. Iyengar, and D. Lamm, Rigorous bounds on the failure of shielding due to helium-ion radiation (2021), in preparation.
[34] C. Biwer, S. Vogel, M. McKerns, and J. Ahrens, Spotlight: Distributed-computing for rietvled analyses using an ensemble of local optimizers (2019), http://github.com/lanl/spotlight.
[35] I. Nakamura, A. Kanemura, T. Nakaso, R. Yamamoto, and T. Fukuhara, Non-standard trajectories found by machine learning for evaporative cooling of 87rb atoms, Opt. Express **27** (2019).
[36] R. Schaback and H. Wendland, Adaptive greedy techniques for approximate solution of large RBF systems, Numerical Algorithms **24**, 239 (2000).
[37] H. Rocha, On the selection of the most adequate radial basis function, Applied Mathematical Modelling **33**, 1573 (2009).
[38] A. S. Dorvlo, J. A. Jervase, and A. Al-Lawati, Solar radiation estimation using artificial neural networks, Applied Energy **71**, 307 (2002).
[39] L. A. Rastrigin, *Systems of External Control* (Mir Publishers, Moscow, 1974) (in Russian).
[40] H. Rosenbrock, An automatic method for finding the greatest or least value of a function, The Computer Journal **3**, 175 (1960).
[41] L. Dixon and G. Szego, The global optimization problem: An introduction, in *Towards Global Optimisation 2* (North-Holland Publishing Company, Amsterdam, 1978) pp. 1–15.
[42] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs* (Springer-Verlag, Berlin, Heidelberg, New York, 1992).
[43] E. Easom, *A Survey of Global Optimization Techniques* (U. of Louisville, Louisville, KY, 1990) m. Eng. Thesis.
[44] D. Lonardoni, I. Tews, S. Gandolfi, and J. Carlson, Nuclear and neutron-star matter from local chiral interactions, Phys. Rev. Research **2**, 022033 (2020).
[45] E. Annala, T. Gorda, A. Kurkela, J. Nättilä, and A. Vuorinen, Evidence for quark-matter cores in massive neutron stars, Nature Physics (2020).
[46] V. Dexheimer, Tabulated neutron star equations of state modelled within the chiral mean field model, Publications of the Astronomical Society of Australia **34**, 10.1017/pasa.2017.61 (2017).
[47] S. Typel, M. Oertel, and T. Klähn, CompOSE CompStar online supernova equations of state harmonising the concert of nuclear physics and astrophysics compose.obspm.fr, Phys. Part. Nucl. **46**, 633 (2015).
[48] M. Hempel, G. Pagliara, and J. Schaffner-Bielich, Conditions for phase equilibrium in supernovae, protoneutron, and neutron stars, Phys. Rev. D **80**, 125014 (2009).
[49] N. K. Glendenning, *Compact Stars: Nuclear Physics, Particle Physics and General Relativity*, Astronomy and Astrophysics Library (Springer New York, 1997).
[50] T. Fischer, I. Sagert, G. Pagliara, M. Hempel, J. Schaffner-Bielich, T. Rauscher, F.-K. Thielemann, R. Käppeli, G. Martínez-Pinedo, and M. Liebendörfer, Core-collapse supernova explosions triggered by a quark-hadron phase transition during the early post-bounce phase, The Astrophys. Journal Supplement Series **194**, 39 (2011).
[51] C. A. Raithel, F. Özel, and D. Psaltis, Finite-temperature Extension for Cold Neutron Star Equations of State, Astrophys. Journal **875**, 12 (2019).
[52] E. Chabanat, P. Bonche, P. Haensel, J. Meyer, and R. Schaeffer, A skyrme parametrization from subnuclear to neutron star densities part ii. nuclei far from stabilities, Nuclear Physics A **635**, 231 (1998).

[53] A. S. Schneider, L. F. Roberts, and C. D. Ott, Open-source nuclear equation of state framework based on the liquid-drop model with skyrme interaction, Phys. Rev. C **96**, 10.1103/physrevc.96.065802 (2017).

[54] J. Stone and P.-G. Reinhard, The skyrme interaction in finite nuclei and nuclear matter, Progress in Particle and Nuclear Physics **58**, 587 (2007).

[55] T. Skyrme, The effective nuclear potential, Nuclear Physics **9**, 615 (1958).

[56] A. Chodos, R. L. Jaffe, K. Johnson, C. B. Thorn, and V. F. Weisskopf, New extended model of hadrons, Phys. Rev. D **9**, 3471 (1974).

[57] S. H. Glenzer and R. Redmer, X-ray Thomson scattering in high energy density plasmas, Reviews of Modern Physics **81**, 1625 (2009).

[58] F. Di Natale, H. Bhatia, T. S. Carpenter, C. Neale, S. Kokkila-Schumacher, T. Oppelstrup, L. Stanton, X. Zhang, S. Sundram, T. R. W. Scogland, G. Dharuman, M. P. Surh, Y. Yang, C. Misale, L. Schneidenbach, C. Costa, C. Kim, B. D'Amora, S. Gnanakaran, D. V. Nissley, F. Streitz, F. C. Lightstone, P.-T. Bremer, J. N. Glosli, and H. I. Ingólfsson, A massively parallel infrastructure for adaptive multiscale simulations: Modeling ras initiation pathway for cancer, in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '19 (Association for Computing Machinery, New York, NY, USA, 2019).

[59] R. Evans, The nature of the liquid-vapour interface and other topics in the statistical mechanics of non-uniform, classical fluids, Advances in Physics **28**, 143 (1979).

[60] S. Ichimaru, Nuclear fusion in dense plasmas, Rev. Mod. Phys. **65**, 255 (1993).

[61] S. D. Baalrud and J. Daligault, Mean force kinetic theory: A convergent kinetic theory for weakly and strongly coupled plasmas, Physics of Plasmas **26**, 082106 (2019), arXiv:1904.09208 [physics.plasm-ph].

[62] A. Diaw and M. S. Murillo, Excess pressure and electric fields in nonideal plasma hydrodynamics, Phys. Rev. E **99**, 063207 (2019).

[63] S. Goldman, An explicit equation for the radial distribution function of a dense lennard-jones fluid, The Journal of Physical Chemistry **83**, 3033 (1979), https://doi.org/10.1021/j100486a020.

[64] E. Matteoli and G. A. Mansoori, A simple expression for radial distribution functions of pure fluids and mixtures, J. Chem. Phys. **103**, 4672 (1995).

[65] A. Morsali, E. K. Goharshadi, G. Ali Mansoori, and M. Abbaspour, An accurate expression for radial distribution function of the Lennard-Jones fluid, Chemical Physics **310**, 11 (2005).

[66] F. J. Rogers, D. A. Young, H. E. Dewitt, and M. Ross, One-component plasma structure factor in tabular form, Phys. Rev. A **28**, 2990 (1983).

[67] J.-L. Bretonnet and A. Derouiche, Analytic form for the one-component plasma structure factor, Phys. Rev. B **38**, 9255 (1988).

[68] N. Desbiens, P. Arnault, and J. Clérouin, Parametrization of pair correlation function and static structure factor of the one component plasma across coupling regimes, Physics of Plasmas **23**, 092120 (2016), arXiv:1606.04675 [physics.plasm-ph].

[69] J. G. Kirkwood, Statistical mechanics of liquid solutions, Chem. Rev. **19**, 275–307 (1936).

[70] M. E. Fisher and B. Wiodm, Decay of Correlations in Linear Systems, J. Chem. Phys. **50**, 3756 (1969).

[71] C. Vega, L. F. Rull, and S. Lago, Location of the fisher-widom line for systems interacting through short-ranged potentials, Phys. Rev. E **51**, 3146 (1995).

[72] P. Hopkins, A. J. Archer, and R. Evans, Asymptotic decay of pair correlations in a yukawa fluid, Phys. Rev. E **71**, 027401 (2005).

[73] S. Plimpton, Fast parallel algorithms for short-range molecular dynamics, J. Comput. Phys. **117**, 1 (1995).

[74] A. Diaw and M. S. Murillo, A DYNAMIC DENSITY FUNCTIONAL THEORY APPROACH TO DIFFUSION IN WHITE DWARFS AND NEUTRON STAR ENVELOPES, The Astrophysical Journal **829**, 16 (2016).

[75] E. B. Bauer and L. Bildsten, Polluted White Dwarfs: Mixing Regions and Diffusion Timescales, Astrophys. J. **872**, 96 (2019), arXiv:1812.09602 [astro-ph.SR].