MSDE

ROYAL SOCIETY
OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue

Check for updates

Cite this: *Mol. Syst. Des. Eng.*, 2021, **6**, 1066

Bayesian optimization of nanoporous materials†

Aryan Deshwal,[a] Cory M. Simon *[b] and Janardhan Rao Doppa*[a]

Received 10th July 2021,
Accepted 6th October 2021

DOI: 10.1039/d1me00093d

rsc.li/molecular-engineering

Nanoporous materials (NPMs) could be used to store, capture, and sense many different gases. Given an adsorption task, we often wish to search a library of NPMs for the one with the optimal adsorption property. The high cost of NPM synthesis and gas adsorption measurements, whether these experiments are in the lab or in a simulation, often precludes exhaustive search. We explain, demonstrate, and advocate Bayesian optimization (BO) to actively search for the optimal NPM in a library of NPMs—and find it using the fewest experiments. The two ingredients of BO are a surrogate model and an acquisition function. The surrogate model is a probabilistic model reflecting our beliefs about the NPM-structure–property relationship based on observations from past experiments. The acquisition function uses the surrogate model to score each NPM according to the utility of picking it for the next experiment. It balances two competing goals: (a) exploitation of our current approximation of the structure–property relationship to pick the NPM we believe [under uncertainty] will be the highest-performing, and (b) exploration of regions of NPM space we have not visited, to pick an NPM we are uncertain about and improve our approximation of the structure–property relationship. We demonstrate BO by searching an open database of ∼70 000 hypothetical covalent organic frameworks (COFs) for the COF with the highest simulated methane deliverable capacity (pertinent for vehicular adsorbed natural gas storage). BO finds the optimal COF and acquires ∼30% of the top 100 highest-ranked COFs after evaluating only ∼140 COFs. More, BO searches more efficiently than evolutionary and one-shot supervised machine learning approaches.

Design, System, Application

Nanoporous materials (NPMs) have many adsorption-based applications. Owing to their tunable structures, we have an abundance of possible NPMs for each adsorption task. Given scarce resources and limited time, how can we *efficiently* search through a library of NPMs to find the optimal NPM for a specific adsorption task? We explain and demonstrate Bayesian optimization (BO) to actively and intelligently search through a library of NPMs to find the optimal one for a given task. BO relies on (i) a continually updated surrogate model to capture our beliefs about the NPM-structure–property relationship and (ii) an acquisition function to make sequential decisions of which NPM to evaluate next, while balancing exploration and exploitation. We show that BO can find the optimal NPM while evaluating only a small fraction of the NPMs in the library of candidates. The adoption of BO for NPM design for specific tasks would impact the discovery and deployment of NPMs by (i) accelerating its schedule, (ii) reducing infrastructure needs, and (iii) lowering costs.

1 Introduction

The selective gas adsorption properties of nanoporous materials (NPMs) endow them with many possible applications in the storage,[1,2] separation,[3] and sensing[4] of gases. As examples, promising applications of NPMs include (i, storage) densifying hydrogen (H_2)—a clean fuel—for compact storage onboard vehicles,[2,5] (ii, separation) capturing carbon dioxide from flue gas of coal-fired power plants; subsequently sequester it to prevent global warming,[6] and (iii, sensing) detecting toxic compounds and explosives.[7,8]

Several classes of NPMs, such as metal–organic frameworks (MOFs),[14] metal–organic polyhedra (MOPs),[15] covalent organic frameworks (COFs),[9] and porous organic cages (POCs),[16] are synthesized modularly by stitching together molecular building blocks *via* coordination (MOFs, MOPs) or covalent (COFs, POCs) bonds to form ∼ crystalline (MOFs, COFs) or molecular (MOPs, POCs) materials. Fig. 1 illustrates the modular synthesis and rational design of COFs in the **sql** topology.[12] The many topologies,[9,17,18] abundance of molecular building blocks, and post-synthetic modifiability[19,20] permit an unlimited number of possible structures exhibiting diverse adsorption properties.

[a] *School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA, USA. E-mail: jana.doppa@wsu.edu*
[b] *School of Chemical, Biological, and Environmental Engineering, Oregon State University, Corvallis, OR, USA. E-mail: cory.simon@oregonstate.edu*

† Electronic supplementary information (ESI) available. See DOI: 10.1039/d1me00093d

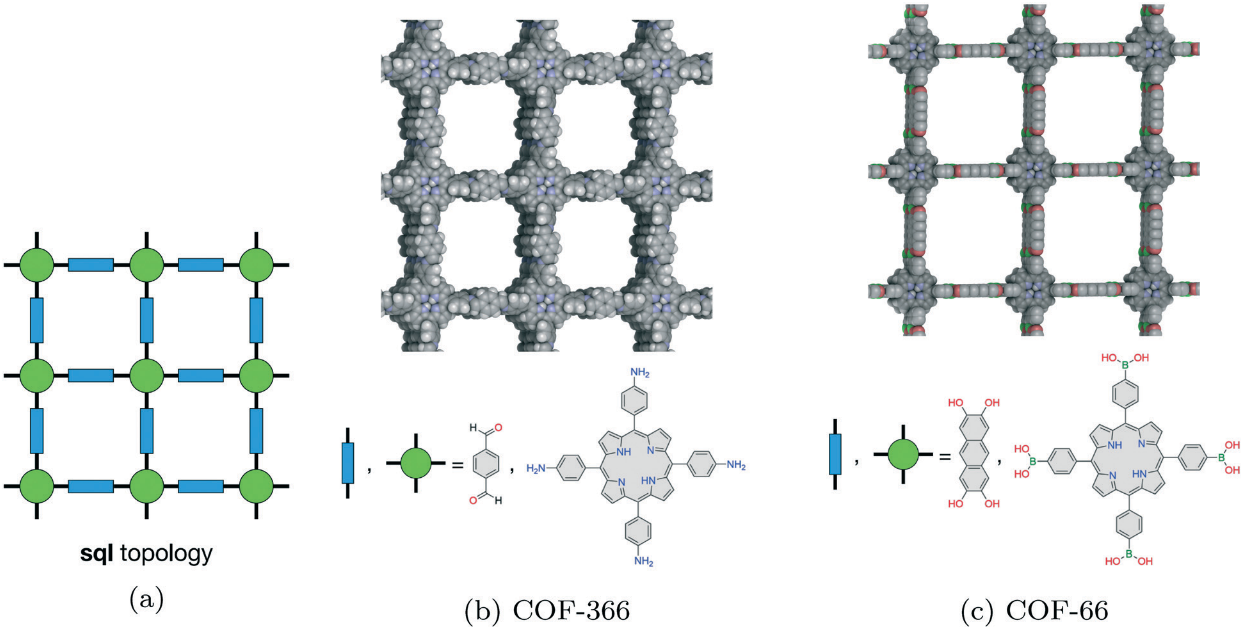


Fig. 1 Illustrating the modular synthesis of covalent organic frameworks (COFs).[9–11] (a) The **sql** (square lattice) network topology specifies the connectivity of tetratopic (green) and diptopic (blue) building units. (b and c) Two examples of COFs in the **sql** topology are (b) COF-366 and (c) COF-66.[12] (Top) The crystal structures. (Bottom) The building blocks: a planar, tetratopic building unit and a linear, diptopic building unit. The building units are stitched together with covalent bonds, through a condensation reaction, to form 2D COF sheets in the **sql** topology. The sheets stack into layers to form 3D channels. Owing to their modular synthesis, on the order of one hundred COFs have been experimentally synthesized and reported.[13]

A common goal is to find, among a large set of candidate NPM structures, the NPM structure(s) with the optimal adsorption property for a given application. As opposed to an exhaustive search, our goal is to search for the optimal NPM *efficiently*, by consuming minimal resources (computational and/or physical) in the process. In the laboratory setting, synthesizing an NPM and measuring its property costs labor and raw materials, and throughput is limited by the capital equipment in the lab. In the computational setting, constructing a high-fidelity computational model of an NPM

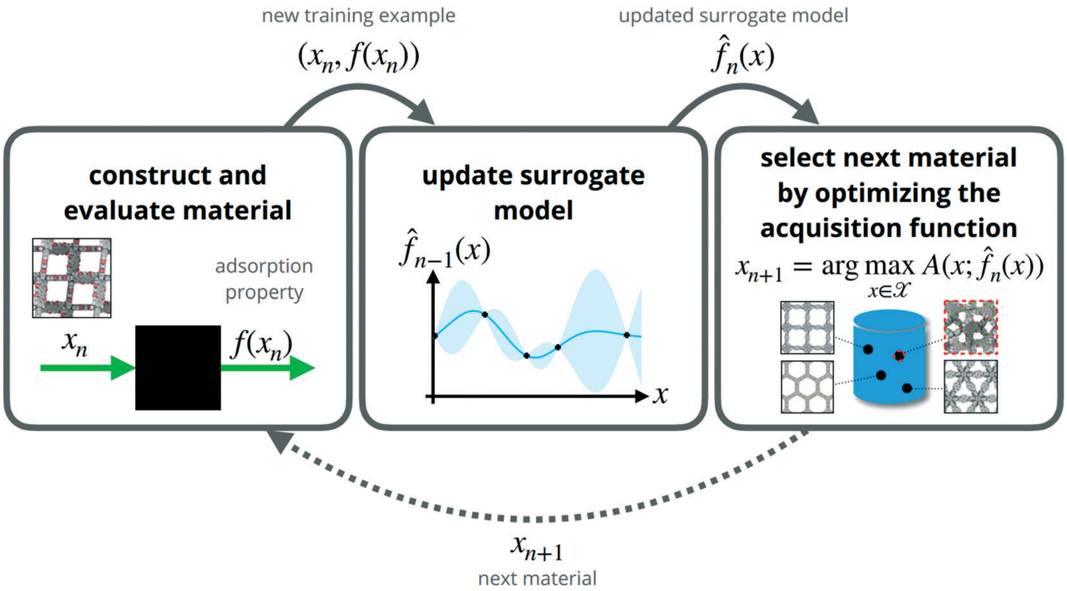


Fig. 2 Bayesian optimization (BO) is an active search method to find the input $\boldsymbol{x}$ that optimizes a black-box objective function $f(\boldsymbol{x})$. In the BO of nanoporous materials, we iterate within a feedback loop: (1) conduct an experiment that measures the property $f(\boldsymbol{x}_n)$ of the material represented by $\boldsymbol{x}_n$. (2) Using the new observation, update our belief about the underlying objective function $f(\boldsymbol{x})$, encapsulated by the surrogate model $\hat{f}(\boldsymbol{x})$. (3) Use the acquisition function $A(\boldsymbol{x})$ to pick the next material $\boldsymbol{x}_{n+1}$ for an experiment.

structure[21–25] and predicting its gas adsorption property through molecular simulations[26,27] consumes electricity, and throughput is limited by computing resources. Thus, both in the laboratory and on the computer, our goal is to find the optimal NPM(s) for a given adsorption task using the fewest experiments (experiment = constructing an NPM and evaluating its adsorption property).

In this article, we explain, demonstrate, and advocate Bayesian optimization (BO) to *actively* and *efficiently* search for NPMs with an optimal property for a given adsorption task. Active, because BO iterates, within a feedback loop, between: (a) conducting an experiment on an NPM, (b) updating our belief about the structure–property relationship, and (c) selecting the NPM for the next experiment. See Fig. 2. Efficient, because BO makes a *data-informed* decision on the NPM to select for the next experiment, while balancing: (a) *exploitation* of our current data-driven belief about the structure–property relationship—to pick an NPM that we believe, under uncertainty, will have the optimal property and (b) *exploration* of regions of the NPM design space where our belief about the structure–property relationship is weak—to pick an NPM we are uncertain about and strengthen our confidence in our approximation of the structure–property relationship.

The two key components of BO are a surrogate model and an acquisition function. The surrogate model, with "surrogate" hinting at "substitute for the experiment"—is a probabilistic model for the structure–property relationship. It is trained on all observations from past experiments. The surrogate model cheaply predicts the properties of the unevaluated NPMs and, critically, quantifies the uncertainty in its predictions. The acquisition function is used to make the decision of which NPM to select for the next experiment. It uses the surrogate model to score the utility of selecting each unevaluated NPM for the next experiment by striking a balance in the exploitation-exploration trade-off. The acquisition function is maximal in regions of the NPM design space where (i) we believe high-performing materials will reside and/or (ii) we are uncertain about the structure–property relationship. The acquisition function relies on the surrogate model to give a resource-efficient, intelligent, active search for optimal NPMs in a wide variety of contexts.

We demonstrate BO of NPMs by efficiently searching an open database[28] of ~70 000 hypothetical COFs, represented by vectors of hand-designed features based on domain knowledge, for those with the highest simulated methane deliverable capacity to store natural gas onboard vehicles.[2] BO recovers the optimal COF(s) using fewer experiments than incumbent strategies including random search, an evolutionary algorithm, and one-shot supervised machine learning using random forests. In the Outlook, we discuss active research areas in BO that are likely to apply to several problems in NPM discovery: (i) batch BO, where experiments are parallelized, (ii) multi-fidelity BO, where NPMs can be evaluated using multiple methods which vary in accuracy and resource cost, (iii) multi-objective BO, where we aim to find a Pareto optimal set of NPMs to optimize multiple properties, and (iv) constrained BO, where our goal is to find high-performing NPMs which can be synthesized.

2 Review of previously used NPM search methods

The NPM research community has adopted several approaches to efficiently search a library of NPMs for the optimal NPM(s).[29,30] We define the efficiency of a search strategy with respect to two naive baselines, where we conduct the high-fidelity experiment on (i, exhaustive search) every NPM in the library, and (ii, random search) a (uniform) random sample of the NPMs in the library.

Supervised machine learning models

A supervised machine learning model can serve as a cheaper, albeit lower fidelity, surrogate for the high-fidelity experiment,[29,31–35] thereby reducing the cost of an exhaustive search. A machine learning approach is predicated upon cheaply computed (relative to the experiment) (i) vector representations of the NPMs—hand-engineered[36,37] or learned[38] —that encode structural features and are correlated to the property or (ii) kernels that capture the tendency for any given pair of NPMs to exhibit similar properties.[39] Training examples are gathered by selecting a small (random or diverse[40,41]) subset of the library of NPMs and labeling them with the property values *via* high-fidelity experiments. Using the training examples, the supervised machine learning model learns to predict the property of any given NPM from *e.g.*, its vector representation. The trained model is then used, as a surrogate for the high-fidelity experiment, to cheaply predict, from their vector representations, the properties of the remaining NPMs in the library. Further high-fidelity experiments may be directed on the NPMs predicted to be optimal by the machine learning model. See ref. 40 and 42–52 as examples.

Genetic algorithms

Genetic algorithms[53] are iterative, stochastic search methods inspired by Darwinian evolution. Each NPM is represented by a "chromosome"—a vector of categorical variables that uniquely specifies its structure. A small initial generation (set) of property-labeled NPMs is iteratively evolved by applying genetic operations to their chromosomes: mutation, replication, selection, and recombination. At each generation, we conduct experiments on each newly evolved NPM to evaluate its fitness. This guides the genetic operations used to evolve to the next generation of chromosomes (representing NPMs), with the ambition of both exploring NPM space and enriching future generations with high-fitness NPMs. See ref. 54–57 as examples.

Monte Carlo tree search

When the NPM search space can be framed as a tree, Monte Carlo tree search[58] is more efficient than random search. Starting at the root node, a policy to select a child node is iteratively applied, giving a path through the tree, culminating at a leaf node. The experiment is then conducted on the NPM represented by the leaf node. Its measured property, viewed as a reward, is back-propagated through the tree to update the statistics of each node along the path to it. Both the visit counts and reward allocations of the nodes are used in the tree policy to balance exploration of new branches of the tree that have not been visited often (or at all) and exploitation of current knowledge to follow branches of the tree that appear likely to lead to optimal NPMs. See ref. 59 and 60 as examples.

Each of these prior approaches suffer from drawbacks. The supervised machine learning approach selects training data to learn a good predictor of the property from the structure representation, then uses the predictor to greedily acquire the NPMs with the highest predicted property. This passive approach can be viewed as one round of exploration and one round of exploitation. Active learning[61] can be used to reduce the number of training examples to learn the structure–property relationship but is not geared towards finding the optimal NPM using the fewest experiments. In BO, we will actively collect training examples according to the goal of finding the optimal NPM with the fewest experiments. Genetic algorithms are also sequential, active search procedures aimed at quickly finding the optimal NPM. However, the genetic operations are heuristic and do not balance exploration and exploration rigorously. As a consequence, genetic algorithms can be difficult to tune and could require many experiments to find the optimal NPMs. MCTS balances exploration and exploitation more rigorously. But, it requires many NPM evaluations to identify promising regions of the tree because it does not explicitly leverage the similarity among structures for principled exploitation. Further, MCTS is limited to NPM design spaces which can be framed as a tree.

3 Problem setup: find the optimal material

Suppose we have a large database of candidate NPM structures, $\mathcal{X}$, for some adsorption task. Let $f:\mathcal{X} \to \mathbb{R}$ be a black-box objective function that, given a candidate NPM $\boldsymbol{x} \in \mathcal{X}$, returns the relevant adsorption property $y = f(\boldsymbol{x})$. Each evaluation of f corresponds to performing an *expensive* experiment—either in the laboratory or in a molecular simulation—to measure the adsorption property y of NPM $\boldsymbol{x}$. Our goal is to find the highest-performing NPM $\boldsymbol{x}^*$ from $\mathcal{X}$ that maximizes the objective function f,

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}\in\mathcal{X}}{\arg\max} f(\boldsymbol{x}), \tag{1}$$

while conducting the fewest number of expensive experiments.

We can interpret $f(\boldsymbol{x})$ as the [unknown] structure–property relationship[62,63] since $\boldsymbol{x}$ [abstractly, at this point in our discussion] represents the structure of a unique NPM and evaluating f means conducting an experiment to measure its property, y.

4 Overview of Bayesian optimization

We explain the key ideas behind the Bayesian optimization (BO) framework[64,65] to find the highest-performing NPM—solve the problem in eqn (1)—*efficiently*, by using the fewest (expensive) experiments.

4.1 Defining an NPM feature space

While $\boldsymbol{x}$ as an abstract representation of an NPM suffices for defining the problem in eqn (1), for BO we must concretely define a NPM *feature space* or *search space* in which each NPM $\boldsymbol{x}$ lies.

Take $\boldsymbol{x}$ to be a fixed-size (among all NPMs) vector representation of the NPM that lies in a continuous feature space. The NPM feature vectors $\{\boldsymbol{x}_i\}_{i=1}^{|\mathcal{X}|}$ should be designed to (1) encode the relevant structural and chemical features of the NPMs, (2) be rotation-, translation-, and, if the NPM is a crystal, replication-invariant, (3) be cheap to compute compared to conducting the experiment (evaluating f), and (4) ideally, each correspond to a unique NPM (injective mapping NPM → $\boldsymbol{x}$).[66] As a result of (1), we expect NPMs close in the feature space to exhibit close values of the property y.

The simplest example of a representation $\boldsymbol{x}$ of an NPM is a list of hand-designed, based on domain expertise, descriptors/features of its structure and composition, such as pore volume, surface area, largest included sphere diameter, density, weight fraction carbon, *etc.*[67–69] Alternatively, $\boldsymbol{x}$ could be learned from a graph[70–72] or 3D image representation[73,74] of a NPM by a graph neural network[38] or convolutional neural network,[75] respectively. See reviews in ref. 29 and 76 for defining feature vectors of NPMs and ref. 40, 42, 44, 52 and 77–80 for different examples of NPM feature spaces. As opposed to dwelling on how to *define* a good feature space of NPMs, we will instead focus on BO, a technique to *search* the defined feature space for the optimal NPM $\boldsymbol{x}^*$ in an efficient manner.

4.2 Active search: exploitation *vs.* exploration

Even with an NPM feature space defined, in practice, the structure–property relationship $f(\boldsymbol{x})$ is a black-box function; analytical expressions for $f(\boldsymbol{x})$ and/or its gradient $\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$ are not known, and it may be multi-modal.

BO is a derivative-free method to *actively* and *efficiently* search the database of NPMs $\mathcal{X}$ for the NPM $\boldsymbol{x}^*$ that maximizes $f(\boldsymbol{x})$. Active, because BO sequentially selects NPMs from $\mathcal{X}$ for experimentation (to evaluate with f), iterating between conducting an experiment and making a decision about which experiment to conduct next. Efficient, because BO makes a data-driven decision to select the next NPM for an experiment while taking into account all observed (NPM $\boldsymbol{x}$, property $y = f(\boldsymbol{x})$) pairs from previous experiments. Each decision to select the next unevaluated NPM from $\mathcal{X}$ to evaluate with f must trade off two conflicting goals:

1) *Exploitation* suggests to use our current, but uncertain, approximation of the structure–property relationship, based on the past observations, to select the NPM that appears to have the most promise as a high-performing material.

2) *Exploration* suggests to select the NPM that we are most uncertain about to improve our approximation of the structure–property relationship.

So, to balance exploitation and exploration, we must balance visits to regions of NPM feature space that (i) appear to contain high-performing NPMs and (ii) have not been explored well. A colloquial example of the exploitation–exploration dilemma in our lives is, in aiming to maximize our enjoyment of food, whether to choose a restaurant that we have visited and know we like versus a new one.[81]

4.3 The ingredients of BO for data-driven decision-making: a surrogate model and an acquisition function

In the BO framework, the two key components used to make each sequential decision of which NPM to conduct an experiment on next are (1) a *surrogate model* that captures our beliefs, based on past observations, about the structure–property relationship and (2) an *acquisition function* that scores each NPM according to the utility of conducting the experiment on it next. The acquisition function uses the surrogate model of the structure property-relationship $f(\boldsymbol{x})$ to decide which NPM to evaluate next while striking a balance between exploitation and exploration.

The surrogate model. The surrogate model is a probabilistic model of the structure–property relationship $f(\boldsymbol{x})$ trained on all observations‡ $\{(\boldsymbol{x}_i, y_i = f(\boldsymbol{x}_i))\}_{i=1}^{n}$ from past experiments. Typically, adopting a Bayesian perspective, the surrogate model treats $f(\boldsymbol{x})$ as a random variable that follows a Gaussian distribution:

$$f(\boldsymbol{x}) \sim \mathcal{N}\left(\hat{y}(\boldsymbol{x}), \sigma^2(\boldsymbol{x})\right) \tag{2}$$

with mean $\hat{y} \in \mathbb{R}$ and variance $\sigma^2 \in \mathbb{R}$. The surrogate model reflects our current beliefs about $f(\boldsymbol{x})$ and serves two purposes in BO. First, to guide exploitation, $\hat{y}(\boldsymbol{x})$ is a cheap-to-evaluate approximation of the expensive-to-evaluate objective function $f(\boldsymbol{x})$, allowing us to cheaply estimate the properties of all unevaluated NPMs. Second, to guide exploration, $\sigma^2(\boldsymbol{x})$ quantifies the uncertainties in the predicted properties of the unevaluated NPMs. This makes us aware of regions in NPM feature space we need to explore to improve our approximation $\hat{y}(\boldsymbol{x})$ and reduce the uncertainty in our beliefs about $f(\boldsymbol{x})$.

The surrogate model is updated in each iteration of BO, after the new observation $(\boldsymbol{x}_{n+1}, y_{n+1} = f(\boldsymbol{x}_{n+1}))$ is gathered, to (i) improve the approximation of $f(\boldsymbol{x})$ and (ii) account for the reduced uncertainty in the region of the feature space surrounding the newly evaluated NPM $\boldsymbol{x}_{n+1}$. Consequently, let us denote the surrogate model after iteration n of BO as $\hat{f}_n$: $\boldsymbol{x} \rightarrow (\hat{y}, \sigma)$.

The acquisition function. The *acquisition function* $A(\boldsymbol{x}; \hat{f}(\boldsymbol{x}))$: $\mathcal{X} \mapsto \mathbb{R}$ scores the utility of, next, evaluating NPM $\boldsymbol{x} \in \mathcal{X}$ with the expensive objective function f. Here, "utility" is defined in terms of our ultimate goal of finding the optimal NPM $\boldsymbol{x}^*$ in eqn (1) with the fewest experiments. The acquisition function employs the prediction of the property $\hat{y}$ and the associated uncertainty σ^2 from the surrogate model $\hat{f}(\boldsymbol{x})$ to assign a utility score to the NPM that balances exploitation and exploration, respectively. Maxima of the acquisition function are located in regions of NPM feature space where the predicted property is large and/or uncertainty is high.

The decision of which NPM to evaluate next is made by maximizing the acquisition function:

$$\boldsymbol{x}_{n+1} = \underset{\boldsymbol{x} \in \mathcal{X}/\mathcal{X}_n}{\arg\max} A\left(\boldsymbol{x}; \hat{f}_n(\boldsymbol{x})\right), \tag{3}$$

where $\mathcal{X}_n$: = $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ is the acquired set of n NPMs that have been evaluated already. Importantly, the acquisition function must be cheap to evaluate.

4.4 Summarizing: BO active search iterations

Fig. 2 illustrates an iteration of BO. At the beginning of iteration n, we conduct an experiment on NPM $\boldsymbol{x}_n$, *i.e.*, we evaluate NPM $\boldsymbol{x}_n$ with the objective function f to obtain a new observation $(\boldsymbol{x}_n, y_n = f(\boldsymbol{x}_n))$. Next, we update the old surrogate model $\hat{f}_{n-1}(\boldsymbol{x})$ to account for this new observation, giving the new surrogate model $\hat{f}_\text{n}(\boldsymbol{x})$. We then select the next (unevaluated) NPM to evaluate, $\boldsymbol{x}_{n+1}$, as the one that maximizes the acquisition function $A(\boldsymbol{x}; \hat{f}_n(\boldsymbol{x}))$.

We terminate BO after we either (i) expend our budget for experiments or (ii) find a material with a satisfactory property value. The BO solution to the problem in eqn (1), $\boldsymbol{x}^*$, then follows from the evaluated NPM with the highest observed property, arg $\max_{i=1}^{N} y_i$, where N is the number of BO iterations (= experiments) performed. Some theoretical work focuses on characterizing how, under specific assumptions, the quality of the approximate optimum in BO scales with the number of iterations.[82]

N.b., typically, the surrogate model is retrained from scratch after each iteration of BO, but some surrogate models can be trained online,[83] reducing the computational cost of the search.

4.5 Remark: BO *vs.* active learning

We remark on a distinction between active learning[84] and Bayesian optimization. Both sequentially collect training

‡ Without loss of generality, the observations are assumed noise-less for clarity of presentation.

examples for a supervised machine learning model. In active learning, the examples are efficiently collected with the goal of reducing the uncertainty in the machine learning model. In Bayesian optimization, the examples are efficiently collected with the goal of, instead, finding the optimal material. BO is more efficient for finding the optimal material than active learning because it avoids collecting examples in regions of feature space that contain poor-performing materials, whereas active learning will do so to reduce the uncertainty of the model in those regions. *E.g.* active learning can be used to sequentially decide pressures at which to conduct molecular simulations of adsorption in an NPM to characterize its adsorption isotherm.[170]

5 Surrogate models and acquisition functions

In this section, we explain surrogate models and acquisition functions commonly used in BO.

5.1 Surrogate models: Gaussian processes

Gaussian processes (GPs)[85,86] are the most commonly used surrogate models in BO owing to their flexibility as function approximators, principled uncertainty quantification, and compatibility with the kernel trick. GPs are non-parametric models that can approximate complicated objective functions $f(\boldsymbol{x})$ given labeled training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$. Through a Bayesian probabilistic framework, GPs provide uncertainty estimates in their predictions and allow incorporation of prior beliefs. GPs rely on a kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$: $\mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ (ref. 87) to capture the similarity between any two NPMs $\boldsymbol{x}$ and $\boldsymbol{x}'$. This gives GPs the flexibility to approximate arbitrary, complicated (but well-behaved!) functions $f(\boldsymbol{x})$. Moreover, it gives GPs versatility in how to represent the NPMs, *e.g.*, graph kernels[88] can be used for NPMs represented as crystal graphs (*e.g.*, ref. 39).

What is a GP?. A GP is a stochastic process that treats the value of the objective function at any given point in feature space, $f(\boldsymbol{x})$, as a random variable. Specifically, GPs assume the joint distribution of any finite collection of function values, say at points $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ on its domain, follows a multi-variate Gaussian distribution

$$\boldsymbol{f} \equiv [f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_m)]^T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}) \tag{4}$$

whose covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$ is given by the kernel function applied pairwise over the points $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$, $\Sigma_{i,j}$: $= k(\boldsymbol{x}_\mathrm{i}, \boldsymbol{x}_\mathrm{j})$. The kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$ quantifies the similarity of NPMs $\boldsymbol{x}$ and $\boldsymbol{x}'$; hence, the idea in GPs is that the properties $f(\boldsymbol{x})$ and $f(\boldsymbol{x}')$ of similar (dissimilar) NPMs $\boldsymbol{x}$ and $\boldsymbol{x}'$ are highly (un)correlated.[89] GPs effectively model the entire function $f(\boldsymbol{x})$—in a point-wise manner—by assuming eqn (4) holds for *any* arbitrary, finite collection of function values on its domain. The mean of zero in eqn (4) assumes the measurements are centered.

From a Bayesian perspective, eqn (4) is a prior assumption about the structure–property relationship $f(\boldsymbol{x})$. When we gather new observations, we will update this prior assumption to arrive at the posterior distribution reflecting our beliefs about the structure–property relationships in light of new data.

Kernel functions. Examples of kernel functions that operate on vector representations of two NPMs $\boldsymbol{x}$ and $\boldsymbol{x}'$ include the linear, polynomial, and radial basis function (RBF) kernels:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f \boldsymbol{x}^T \boldsymbol{x}' \quad \text{linear kernel} \tag{5}$$

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f (\boldsymbol{x}^T \boldsymbol{x}')^d \quad \text{homogeneous polynomial kernel} \tag{6}$$

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f e^{-||\boldsymbol{x}-\boldsymbol{x}'||_2^2/(2\gamma^2)}. \quad \text{radial basis function (RBF) kernel} \tag{7}$$

Each kernel possesses the hyperparameter σ_f, the signal variance, which is a scale factor controlling the expected range of the functions represented by the GP. The polynomial kernel has a hyperparameter d that controls the order of the polynomial in the features, and the RBF kernel contains a length-scale hyperparameter γ that controls how close $\boldsymbol{x}$ and $\boldsymbol{x}'$ must be in the feature space to be considered "similar" and the expected roughness of the functions represented by the GP. Implicitly, each nonlinear kernel maps the two vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ into a new, higher-dimensional feature space through a mapping τ, then takes the inner product of the vectors in the new feature space:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \tau(\boldsymbol{x})^T \tau(\boldsymbol{x}'). \tag{8}$$

Interestingly, the feature map $\tau(\boldsymbol{x})$ corresponding to the RBF kernel in eqn (7) maps vectors into an *infinite* dimensional feature space! By implicitly operating in a higher-dimensional feature space, nonlinear kernels give GPs more flexibility, or expressiveness, for approximating complicated objective functions $f(\boldsymbol{x})$. Notably, graph kernels[88] and image kernels[90] can define similarities of two NPMs represented as graphs (nodes: atoms, edges: bonds) and images, respectively.

Inference with GPs. In BO, we exploit GPs for regression, with uncertainty quantification, to build a surrogate model for $f(\boldsymbol{x})$. We have observations $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$ from previous experiments (previous iterations of BO), with y_i the measured property of NPM $\boldsymbol{x}_i$. Under the Bayesian view, y_i is a noise-free observation§ of the random variable $f(\boldsymbol{x}_i)$. We wish to know the distribution of the random variable $f(\boldsymbol{x})$ for an *unevaluated* NPM $\boldsymbol{x}$, to determine the utility of evaluating it in the next experiment. Imposing the assumption in eqn (4) for a specific collection of points on the domain of $f(\boldsymbol{x})$ composed of (i) the n evaluated NPMs from the past experiments $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ and (ii) the unevaluated NPM, $\boldsymbol{x}$:

$$\begin{bmatrix} \boldsymbol{f} \\ f(\boldsymbol{x}) \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\sigma} \\ \boldsymbol{\sigma}^T & k(\boldsymbol{x}, \boldsymbol{x}) \end{bmatrix}\right) \tag{9}$$

§ Note that we can pose GPs that relax the assumption that the observations are noise-free.[85]

with $\boldsymbol{f} = [f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_n)]^T$ the vector of random variables representing the properties of the previously evaluated NPMs, $\boldsymbol{\sigma} = [k(\boldsymbol{x}, \boldsymbol{x}_1), k(\boldsymbol{x}, \boldsymbol{x}_2), \ldots, k(\boldsymbol{x}, \boldsymbol{x}_n)]^T$ the vector of the kernel between the unevaluted NPM and the previously evaluated NPMs, and $\boldsymbol{\Sigma}$ the $n \times n$ kernel matrix for the previously evaluated NPMs, with $\Sigma_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. However, we have observations of the random variables in $\boldsymbol{f}$, $\boldsymbol{y} = [y_1, y_2, \ldots, y_n]^T$. Conditioning the joint distribution in eqn (9) on the observations $\boldsymbol{y}$ of the random variables $\boldsymbol{f}$, we arrive at the posterior distribution for the property $f(\boldsymbol{x})$ of the *unevaluated* NPM, also Gaussian:

$$f(\boldsymbol{x})|\boldsymbol{y} \sim \mathcal{N}(\hat{y}(\boldsymbol{x}), \sigma^2(\boldsymbol{x})), \tag{10}$$

with mean and variance:

$$\hat{y}(\boldsymbol{x}) = \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{y} \tag{11}$$

$$\sigma^2(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}. \tag{12}$$

We can interpret eqn (11) and (12). The predicted property of the unevaluated NPM, $\hat{y}(\boldsymbol{x})$, is a linear combination of the observed properties of the evaluated NPMs, $\boldsymbol{y}$, with weights $\boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}$. The weight on each measured property depends on the similarity between that NPM and the unevaluated NPM. The variance $\sigma^2(\boldsymbol{x})$ describing the uncertainty associated with the prediction of the property of the unevaluated NPM $\boldsymbol{x}$ is the prior assumption of $k(\boldsymbol{x}, \boldsymbol{x})$ reduced by $\boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}$, which captures the similarity of the unevaluated NPM $\boldsymbol{x}$ with the set of previously evaluated NPMs.

Fig. 3 illustrates a GP model of a toy function $f(\boldsymbol{x})$, based on an RBF kernel, over a toy one-dimensional NPM feature space $\mathcal{X} \subset \mathbb{R}$, trained on five observations. The mean in the GP, $\hat{y}(\boldsymbol{x})$, approximates $f(\boldsymbol{x})$, and the variance $\sigma^2(\boldsymbol{x})$ expresses uncertainty in the approximation. Generally, the uncertainty is small close to an observation and large when far from an observation.

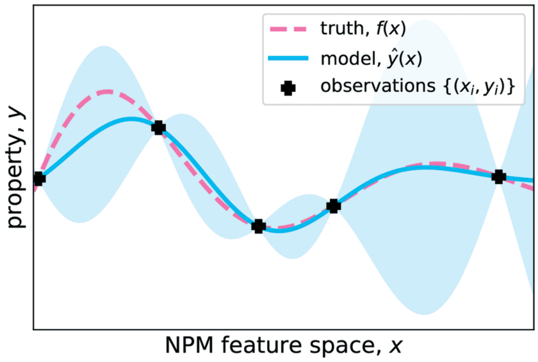


Fig. 3 Illustration of a Gaussian process (GP) model with an RBF kernel over a toy one-dimensional NPM feature space. The black points are the observed data from a toy structure–property relationship, $f(x)$. The blue line and shaded region visualize the GP model trained on the observed data: the line is the approximation $\hat{y}(x)$ of the structure–property relationship, while the shaded region illustrates the uncertainty by covering $y(x) \pm 2\sigma(x)$. The GP model shows large (small) uncertainty in regions of feature space far from (close to) the observations.

Hyperparameters of GPs. GPs are non-parametric models, but most useful kernel functions used in GPs contain hyperparameters. For example, the RBF kernel in eqn (7) has the length-scale γ and the signal variance σ_f hyperparameters. To learn the hyperparameters of the kernel that give us the best approximation of $f(\boldsymbol{x})$, typically we maximize the marginal likelihood of the observed data as a function of the kernel hyperparameters.[89] Generally, at each iteration of BO, the hyperparameters of the kernel are updated to account for the newly acquired observation.

Two further interpretations of GP models of functions. A GP model of the objective function $f(\boldsymbol{x})$ can be interpreted as (i, weight space view) Bayesian linear regression in the implicit feature space of the kernel and (ii, function space view) a distribution over functions.[85] To clarify, the GP model of $f(\boldsymbol{x})$ in eqn (4) is equivalent to the parametric model:

$$f(\boldsymbol{x}) \sim \boldsymbol{w}^T\tau(\boldsymbol{x}), \tag{13}$$

with weights $\boldsymbol{w}$ on which we place a Gaussian prior and τ the map associated with the kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ used in the GP. In the weight space view, GP inference models the posterior distribution over weights $\boldsymbol{w}$ in eqn (13). In the function space view, GP inference models the posterior distribution over the space of functions represented in eqn (13). Though eqn (13) is helpful for understanding GPs and sampling functions from the distribution over the function space they describe, we in practice conduct GP inference using the kernel, through eqn (11) and (12). *E.g.* $\tau(\boldsymbol{x})$ is a vector of infinite dimension in the case of the RBF kernel, making the view of GPs in eqn (13) unfriendly for computations.

5.2 Examples of acquisition functions

We provide three common examples of acquisition functions and explain how they use the surrogate model to, while trading exploration and exploitation, select the NPM to evaluate in the next experiment.

Upper confidence bound (UCB). The UCB acquisition function selects the point that maximizes the upper confidence function:

$$A(\boldsymbol{x}) := \hat{y}(\boldsymbol{x}) + \beta\sigma(\boldsymbol{x}) \tag{14}$$

where $\hat{y}(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ are the predicted property of NPM $\boldsymbol{x}$ and its associated uncertainty, respectively, provided by the surrogate model. The parameter β explicitly trades exploration and exploitation: if β is large (small), the UCB is exploratory (exploitative) and tends to select NPMs with the highest uncertainty (predicted property). To explain the terminology of UCB, the top boundary of the shaded region that bands $\hat{y}(\boldsymbol{x})$ in Fig. 3 is the UCB for $\beta = 2$.

Expected improvement (EI). Another acquisition strategy is to select the NPM with the highest expected improvement

(EI) of the property of the best evaluated NPM so far. Let the random variable $I(\boldsymbol{x})$: = max(0, $f(\boldsymbol{x})$ − $\max_i y_i$) denote the improvement in the property of a NPM $\boldsymbol{x}$ over the best observed NPM thus far. The EI is then:

$$A(\boldsymbol{x}) := \int_{-\infty}^{\infty} I(\boldsymbol{x})\mathcal{N}\left(y|\hat{y}(\boldsymbol{x}), \sigma^2(\boldsymbol{x})\right) dy, \tag{15}$$

which can be written in closed form:

$$A(\boldsymbol{x}) = \begin{cases} (\hat{y}(\boldsymbol{x}) - \max_i y_i)\Phi\left(\dfrac{\hat{y}(\boldsymbol{x}) - \max_i y_i}{\sigma(\boldsymbol{x})}\right) + \sigma(\boldsymbol{x})\phi\left(\dfrac{\hat{y}(\boldsymbol{x}) - \max_i y_i}{\sigma(\boldsymbol{x})}\right) & \sigma(\boldsymbol{x}) > 0 \\ 0 & \sigma(\boldsymbol{x}) = 0, \end{cases} \tag{16}$$

with Φ and ϕ the cumulative and probability distribution functions, respectively, of the standard normal distribution. The first and second terms in eqn (16), respectively, capture the exploitation and exploration component of EI.

Information-theoretic acquisition functions. The principle behind acquisition functions based on information theory is to select the NPM $\boldsymbol{x}$ that maximizes the mutual information between (i) its property $y = f(\boldsymbol{x})$ and (ii) the location of the NPM $\boldsymbol{x}^*$ in feature space that maximizes $f(\boldsymbol{x})$. Viewing both $f(\boldsymbol{x})$ and $\boldsymbol{x}^*$ as random variables, the following acquisition function describes the information gain about the location $\boldsymbol{x}^*$ of the optimal NPM by observing the property $y = f(\boldsymbol{x})$ of a newly acquired NPM, $\boldsymbol{x}$.

$$A(\boldsymbol{x}) = \mathrm{MI}[(\boldsymbol{x}, y), \boldsymbol{x}^*] \tag{17}$$

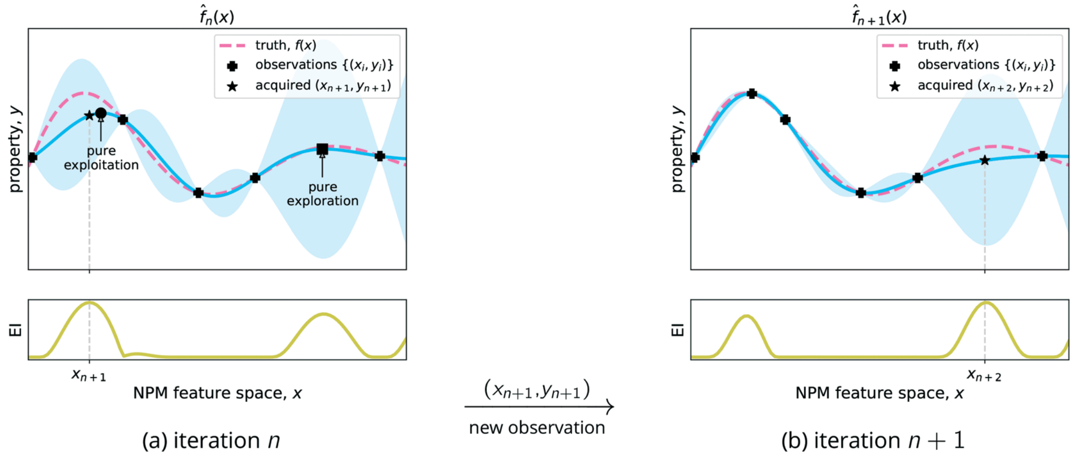
$$= H[p(\boldsymbol{x}^*|D_n)] - \mathrm{E}_y[H[p(\boldsymbol{x}^*|D_n \cup (\boldsymbol{x}, y))]] \tag{18}$$

where MI[·,·] is the mutual information between two random variables, $\mathrm{E}_y[\cdot]$ the expectation over y, $D_n = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ the observations thus far, and $H(\cdot)$ is the entropy of a probability density function $p(\cdot)$. The mutual information is the reduction in the entropy of the probability density function of the location of the optimum NPM, $\boldsymbol{x}^*$, as a result of observing the property $y = f(\boldsymbol{x})$ of NPM $\boldsymbol{x}$. The distribution $p(\boldsymbol{x}^*)$ could be approximated under a GP surrogate model by sampling functions from eqn (13) then optimizing them. In practice, eqn (18) is expensive to compute, but there are several acquisition functions based on instantiations of this general idea.[91,169]

Illustrating BO acquisition and the exploration–exploitation tradeoff. Fig. 4a illustrates the EI acquisition function in a toy one-dimensional NPM space under a GP surrogate model with n = 5 observations. The EI acquisition function exhibits two maxima. The first (global) maximum is in a region of the feature space where the predicted property $\hat{y}(\boldsymbol{x})$ is the largest. The second (local) maximum is where the uncertainty $\sigma(\boldsymbol{x})$ is largest. We select the NPM for the next experiment, x_{n+1}, as the one that maximizes the EI acquisition function. Fig. 4a shows the acquired NPM assuming the database of NPMs $\mathcal{X}$ covers all points on the domain shown. To illustrate how the EI acquisition strategy balances exploration and exploitation, for comparison, we also show the acquired NPM $\boldsymbol{x}_{n+1}$ if the acquisition strategy were purely exploration and purely exploitation. Pure exploration dictates $\boldsymbol{x}_{n+1}$ = arg max $\sigma(\boldsymbol{x})$, but this NPM has a poor property. Pure exploitation dictates $\boldsymbol{x}_{n+1}$ = arg max $\hat{y}(\boldsymbol{x})$, but this NPM is too close to an existing observation. EI balances the trade-off by picking an NPM with both a high uncertainty and high predicted property.

6 Experiments and results

We now demonstrate Bayesian optimization by using it to efficiently search for covalent organic frameworks (COFs) for vehicular natural gas storage.[2] Our experiments below use the open data from Mercado *et al.*[28] and can be fully reproduced on a desktop computer using our computer code at github.com/aryandeshwal/BO_of_COFs.



(a) iteration n

(b) iteration $n + 1$

Fig. 4 Illustration of one iteration of BO where (i) the acquisition function is used to select the next NPM whilst balancing exploitation and exploration and (ii) the GP surrogate model based on the RBF kernel is updated to account for the newly acquired observation. (a) Iteration n. (b) Iteration n + 1, after the surrogate model is updated by the new observation (x_{n+1}, y_{n+1}) acquired to maximize the expected improvement (EI). In both (a) and (b), the top panel shows the surrogate model, and the bottom panel shows the expected improvement (EI) acquisition function. For comparison, in (a) we illustrate the NPM x_{n+1} that would have been selected under a pure exploitation or pure exploration acquisition strategy.

6.1 Experimental problem setup

Our goal is to efficiently search a database of COFs for the one with the largest methane deliverable capacity.

The database of COFs, $\mathcal{X}$. The database of COFs contains 69 840 2D and 3D predicted COF structures constructed by Mercado *et al.*[28]

The COF vector representation, $\boldsymbol{x}$. We represent each COF structure with a vector $\boldsymbol{x} \in \mathbb{R}^{12}$ of structural and chemical features listed in Table 1 and computed by Mercado *et al.*[28] We min–max normalized each feature¶ to lie in [0, 1]. This defines COF feature space as $[0, 1]^{12}$.

The methane deliverable capacity, y. The COF property we wish to maximize is the simulated deliverable capacity of methane [L STP CH_4/L COF] at 298 K under a 65 bar to 5.8 bar pressure swing. The deliverable capacity of the COF primarily determines the driving range of a vehicle on a "full" adsorbed natural gas fuel tank packed with the COF.[2]

The expensive objective function, $f(\boldsymbol{x})$. Evaluating the objective function f to give $y = f(\boldsymbol{x})$ involves conducting two grand-canonical Monte Carlo simulations of methane adsorption in the COF structure represented by $\boldsymbol{x}$—one at (65 bar, 298 K) and one at (5.8 bar, 298 K). The deliverable capacity y then follows from the difference in the simulated methane adsorption at the two conditions. The function f is expensive to evaluate, as the run time of the molecular simulations is on the order of hours.

Goal: data-efficient search for the optimal COF, $\boldsymbol{x}^*$. In an *exhaustive search*, we would conduct expensive molecular simulations to predict the methane deliverable capacity of each candidate COF in the database—*i.e.*, collect $\{(\boldsymbol{x}, y = f(\boldsymbol{x})) : \boldsymbol{x} \in \mathcal{X}\}$—to find the COF $\boldsymbol{x}^* \in \mathcal{X}$ with the highest deliverable capacity. In contrast to an exhaustive search, instead, our goal is to find the optimal COF $\boldsymbol{x}^*$ *efficiently*–while conducting expensive molecular simulations *in only a small proportion of the candidate COFs*.

We hypothesize that BO will provide a simulation-efficient search for the optimal COF, $\boldsymbol{x}^*$. In reality, Mercado *et al.*[28] already simulated methane adsorption in all of the COFs at (65 bar, 298 K) and (5.8 bar, 298 K) and computed their methane deliverable capacities. Thus, (i) we know the optimal COF $\boldsymbol{x}^*$ and (ii) as opposed to actually conducting molecular simulations of methane adsorption in a selected COF during the active search, we instead look up the result of the simulations (the deliverable capacity) from the data of Mercado *et al.*[28] Each data lookup, conceptually, represents conducting the two expensive molecular simulations of methane adsorption in a COF ourselves. N.b., that we look up data as opposed to conducting a simulation ourselves has no impact on the BO search efficiency when defined in terms of the number of molecular simulations needed to find the optimal COF. The exhaustive search of Mercado *et al.*[28] allows us to readily evaluate the simulation-efficiency of different search strategies to find the optimal COF(s). We will compare the search efficiency of BO to random search, an evolutionary algorithm, and one-shot supervised learning.

Table 1 Features comprising the vector representation of a COF, $\boldsymbol{x}$, broken into those that capture its structure and chemical composition

Structural (geometrical)	Chemical composition
Void fraction	Density of carbon
Density	Density of flourine
Largest included sphere diameter	Density of hydrogen
Largest free sphere diameter	Density of nitrogen
Gravimetric surface area	Density of oxygen
	Density of sulfur
	Density of silicon

6.2 Search strategies

We use several different strategies to search for the optimal COF $\boldsymbol{x}^*$ exhibiting the highest methane deliverable capacity y in the database $\mathcal{X}$.

Random search. Random search is a naive baseline. At each iteration, we uniform randomly select an unevaluated COF from $\mathcal{X}$ to evaluate. Random search does not make a *data-informed* selection of a COF for the next evaluation, as it ignores the past observations, $\{(\boldsymbol{x}_i, y_i = f(\boldsymbol{x}_i))\}$. Thus, random search is expected to perform poorly.

Bayesian optimization (BO). For BO, we employ (1) a Gaussian process (GP) with the Matérn kernel ($\nu = 2.5$)[85] as our surrogate model and (2) the expected improvement (EI) in eqn (16) as our acquisition function. To initialize the GP surrogate model for BO, we first randomly select ten COFs from the database and evaluate their methane deliverable capacity. Our [loose] reasoning for selecting ten COFs to initialize the GP model for BO was to ~ match the number of COF features. The GP surrogate model is then trained on $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{10}$, which count towards the number of evaluations when we report the search-efficiency of BO. At each iteration of BO, we fit a new GP to all past observations $\{(\boldsymbol{x}_\mathrm{i}, y_\mathrm{i})\}$, which includes choosing the hyperparameters of the Matérn kernel (length-scale and signal variance) by maximizing the marginal likelihood of the data under the GP. We implemented our BO procedure in the BoTorch library.[92] In accordance with the assumption behind GPs, we z-score standardize the deliverable capacities to have mean zero and unit variance (using the training data only). During the acquisition phase, we evaluate the acquisition function for each COF in the database and select the COF with the highest value, in contrast to optimizing the acquisition function over the continuous COF feature space.

Evolutionary search (*via* CMA-ES). As an evolutionary search method, we use covariance matrix adaptation evolution strategy (CMA-ES),[93,94] a state-of-the-art, stochastic optimizer for rugged, non-convex, black-box objective

¶ We used the feature vectors of all COFs for the min–max normalization (both acquired and non-acquired COFs). This does not constitute data leakage because, in our setting, (i) the features are cheap to compute and (ii) we have a finite library of COFs for which it is feasible to compute all features for all COFs in $\mathcal{X}$.

functions. In CMA-ES, new COFs are stochastically selected from the search space by sampling from a multivariate Gaussian distribution over the feature space. The mean and covariance matrix of the distribution are updated over the search process, as COFs are acquired and evaluated in batches (generations). To update the distribution, with the aim of increasing the likelihood of acquiring and taking search steps towards high-performing COFs, (i) the mean is updated using a weighted average of the most high-performing COFs in the generation (a selection mechanism) and (ii) the covariance matrix is updated using a weighted average of the best search steps (from the mean) towards the high-performing COFs.[94]

CMA-ES has two hyperparameters: the initial standard deviation for each COF feature and the number of new candidate COFs acquired in each iteration (the population size). We initialized the CMA-ES algorithm with a randomly selected COF and set the initial standard deviation to 0.5 to cover our COF feature space $[0,1]^{12}$. The population size, 11, was determined by a default heuristic in the cma library in Python.

CMA-ES operates in a continuous search space. When it selects a point in COF feature space for the next generation, it does not exactly correspond to a feature vector of a COF in the database; to apply CMA-ES, we select the nearest (in feature space), unacquired COF in the database.

One-shot supervised learning (*via* RF). While one-shot supervised learning does not constitute active search like BO, it is the most popular COF acquisition method to circumvent an exhaustive search for the optimal NPM using the high-fidelity evaluation method. A one-shot supervised learning strategy progresses through three stages. (1) To explore, we select a small subset of the COFs and evaluate them. This data serves as training examples for the supervised machine learning model. (2) We use the trained model to predict the deliverable capacity of the remaining, unevaluated COFs. (3) Exploiting our trained model, we acquire the top-k unevaluated COFs with the highest predicted deliverable capacities and evaluate them. Stages (1) and (3) both incur (costly) COF evaluations.

We compare one-shot supervised learning with the active search strategies by comparing the deliverable capacities among their acquired sets of COFs when given the same budget of COF evaluations. Much like balancing exploration and exploitation, we elect to split the budget of evaluations for one-shot supervised learning among stages (1) and (3) equally.

For stage (1), we assess two training set acquisition strategies: (i) uniform random selection and (ii) max-min diversity selection[40,95] of COFs. For (ii), we sequentially acquire COFs for the training set: at each iteration, we select the COF with the maximum minimum distance (in COF feature space) to a COF already in the training set. We initialize the diverse set with a random COF.

As the supervised learning model, we use the commonly-used[40–42,46] random forest (RF) regression model (100 trees, default parameters in scikit-learn) to approximate $f(\boldsymbol{x})$ using the (differently sized) training sets.

6.3 Results

We now execute each strategy to search for the optimal COF $\boldsymbol{x}^*$ exhibiting the highest methane deliverable capacity y in the database $\mathcal{X}$.

6.3.1 Search efficiency. Search efficiency curves for BO, in terms of different performance metrics, are shown in Fig. 5 (blue). Each search efficiency curve describes the quality of the acquired set of COFs $\mathcal{X}_n$ as the number of COFs evaluated/acquired, n, (= the number of BO iterations = the number of simulations/"experiments") increases. The metrics of acquisition set quality are (Fig. 5a) the maximum deliverable capacity among the COFs in $\mathcal{X}_n$, (Fig. 5b) the highest deliverable capacity rank—rank among the entire data set $\mathcal{X}$—among the COFs in $\mathcal{X}_n$ and (Fig. 5c) the fraction of the 100 COFs—top 100 in $\mathcal{X}$—with the highest deliverable capacity in $\mathcal{X}_n$. All 100 BO searches acquired the optimal COF after $n = 139$ COF evaluations. After $n = 250$ COF evaluations, BO acquired 36% of the top 100 COFs in the data set of ∼70 000 COFs. This illustrates how BO can provide a simulation-efficient search for the optimal COF, as opposed to conducting an exhaustive search.

N.b., the shaded bands surrounding the search efficiency curves in Fig. 5 show the standard deviation among the 100 runs; the stochasticity for BO emanates from the initialization of the surrogate model, where we acquired ten random COFs (different for each run) to train it.

BO also provides a more experiment-efficient search than random search, evolutionary search, and the one-shot supervised learning approach. Given the same budget of COF evaluations ($n = 250$), a random search (on average) acquires only the 340th ranked COF and 0.21% of the top 100 COFs. The performance of random search is poor because it selects the next COF to evaluate without considering the previous observations. Evolutionary search and one-shot supervised learning (diverse training set) provide a much more efficient search than random search, acquiring the 11th/20th ranked COF and the top 12%/15% of the top 100 COFs on a budget of 250 evaluations. Though, evolutionary search nor the one-shot supervised learning strategy recover the optimal COF $\boldsymbol{x}^*$ after 250 evaluations. Thus, BO outperforms the baseline search methods of evolutionary search and one-shot supervised learning using both metrics of (a) the highest deliverable capacity in the acquired set and (b) the fraction of the top 100 COFs in the acquired set (when given a budget of fewer than 250 evaluations). N.b., BO is designed to optimize the performance metric (a), but BO could be tailored to optimize a top-k metric.[96,97]

Except when the training set is very small (10), the search performance of the one-shot supervised learning strategy *via* a RF benefits from acquiring a diverse training set, compared to a randomly selecting training examples.

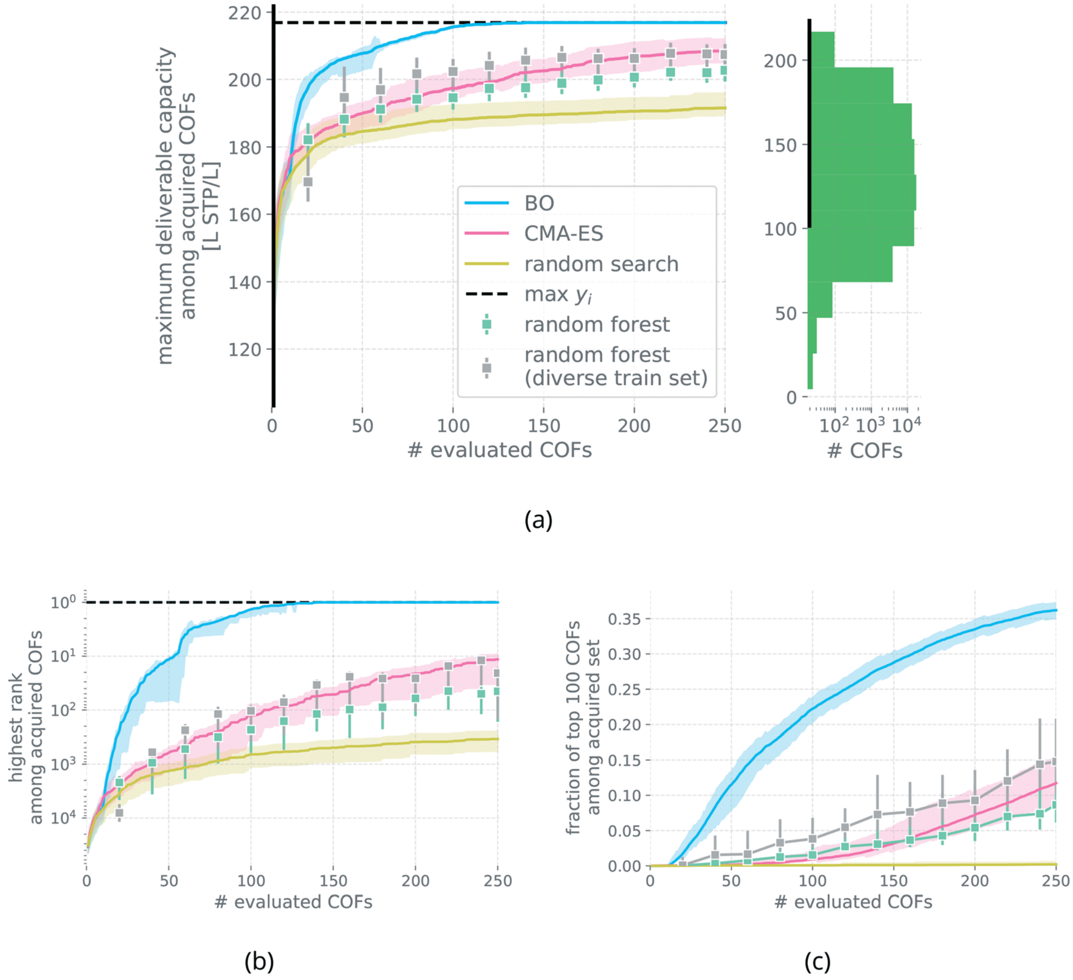


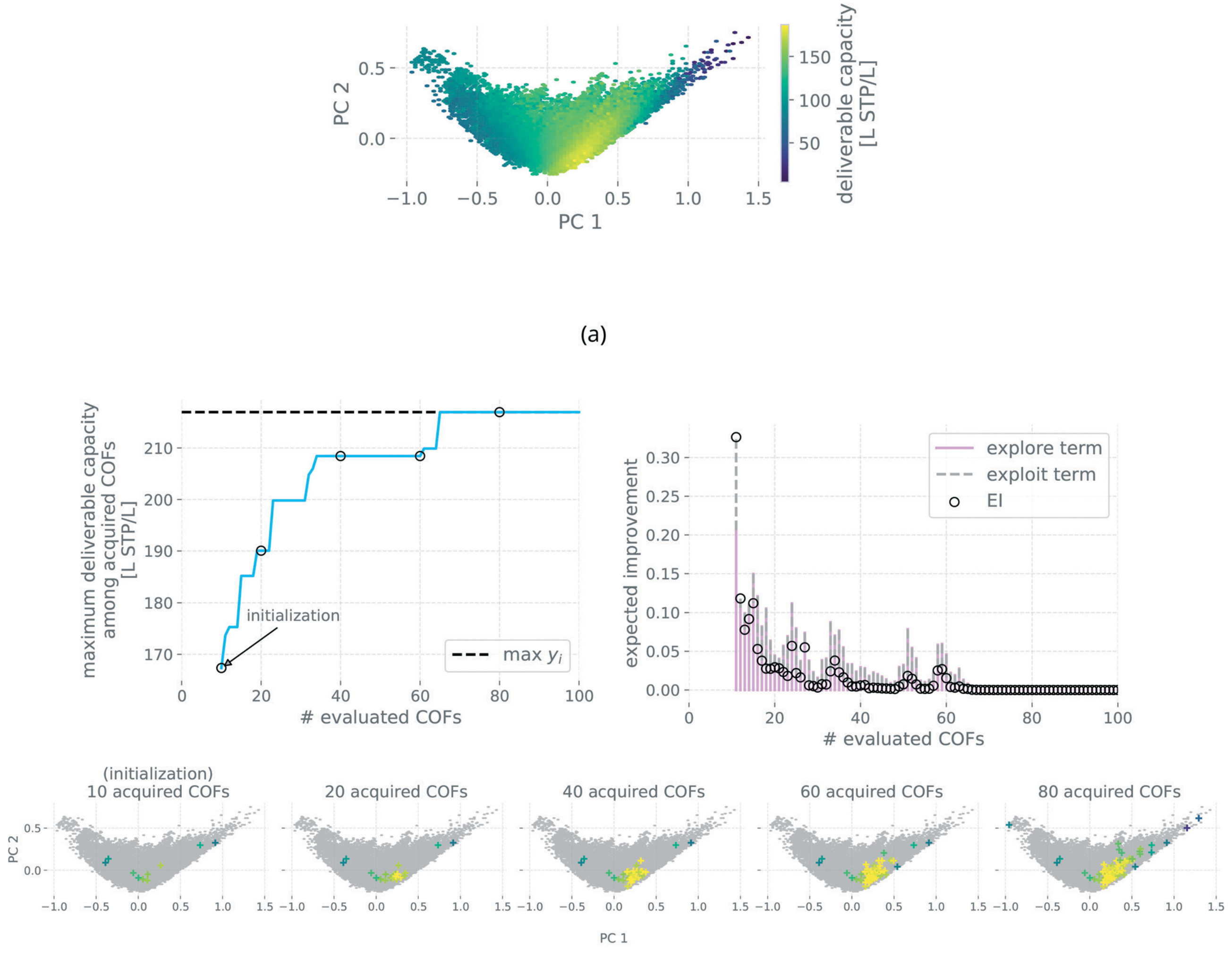
Fig. 5 The search efficiency of BO in comparison with random search, evolutionary search, and one-shot supervised learning. The search efficiency curves show (a) the maximum deliverable capacity, (b) the highest rank (among the entire data set $\mathcal{X}$) of the deliverable capacity, and (c) the fraction of the top 100 ranked (among $\mathcal{X}$) COFs–among the acquired COFs as the number of acquired COFs increases. The shaded region shows the variance over 100 [stochastic] runs. To give (a) context, we show the distribution of the deliverable capacities among the COFs in the entire data set, $\mathcal{X}$, on the right and two black bars to facilitate comparison of the scales on the y-axes.

6.3.2 Visualizing the BO acquisition set. To understand the behavior of BO for searching the database of COFs for the optimal COF $\boldsymbol{x}^*$, we visualize the acquisition set of COFs in feature space as one of the BO searches progresses. Given that the feature space is 12-dimensional, we resort to principal component analysis (PCA) to [approximately] reduce the dimension of the feature space to two. *I.e.*, we project each COF feature vector onto a reduced, 2D feature space through PCA of the data $[\boldsymbol{x}_1\boldsymbol{x}_2\cdots\boldsymbol{x}_{|\mathcal{X}|}]$.

First, we attempt to visualize the structure–property relationship $f(\boldsymbol{x})$. Fig. 6a shows a depiction of $f(\boldsymbol{x})$ as a 2D heatmap over the reduced 2D feature space. The color of each voxel in the reduced COF space indicates the average deliverable capacity of COFs that fall in that voxel.

Fig. 6b shows the acquired set of COFs, colored by deliverable capacity, at 10 (initialization), 20, 40, 60, and 80 iterations of a BO run. For reference, (i) the gray background shows the coverage of COF space by all COFs in the dataset, shown in Fig. 6a and (ii) the top left panel shows the search efficiency curve for this run. The top right panel of Fig. 6b explains the acquisition decisions of BO by showing the value of the EI acquisition function for each acquired COF, broken into contributions from the explorative and exploitative components (see eqn (16)). Early in the search, exploration dominates, while exploitation dominates at the later stages.

6.3.3 Effect of the initial training set size. The strategy to select and size of the training set to initialize the GP surrogate model for BO may impact the search efficiency of BO. We opted to initialize the GP surrogate model for BO with a training set of ten randomly selected then evaluated COFs from the library. Here, we conduct BO searches using varying initial training set sizes (5, 10, 15, 20, 25) to assess the impact on the search efficiency curve of BO. See Fig. S1.† Interestingly, the trend is that BO acquires higher deliverable capacity COFs sooner if initialized with fewer COFs. Another GP initialization strategy for BO is to acquire a single COF closest to the center of the domain.[98]



(a)

(b)

Fig. 6 Interpreting the acquisition behavior of BO. (a) An attempt to visualize *f*(***x***). The plane shows the first two principal components of COF feature space with each voxel colored according to the average deliverable capacity of the COFs falling in it. (b) Pertaining to one BO run: (top left) search efficiency curve, marked by the stages at which we visualize the acquired set of COFs. (Top right) The value of the EI acquisition function (markers) of each acquired COF, partitioned into an exploratory and exploitative (usually negative) component. (Bottom) To visualize the acquired set of COFs, points are the acquired COFs in reduced 2D COF feature space at 10, 20, 40, 60, and 80 iterations and are colored according to the deliverable capacity (color bar in (a) pertains). The gray background shows the region of the feature space covered by the COFs in (a).

6.3.4 Balancing exploration and exploitation. We conceptually illustrated how the expected improvement acquisition function balances exploration and exploitation in Fig. 4. We now show how balancing exploration and exploitation is crucial for BO to recover the optimal COF with the fewest experiments. To do so, we compare the search efficiency of BO using three different acquisition functions:

$$A(\boldsymbol{x}) = \mathrm{EI}(\boldsymbol{x}) \quad \text{exploration–exploitation balance} \tag{19}$$

$$A(\boldsymbol{x}) = \hat{y}(\boldsymbol{x}) \quad \text{pure exploitation} \tag{20}$$

$$A(\boldsymbol{x}) = \sigma(\boldsymbol{x}) \quad \text{pure exploration} \tag{21}$$

The expected improvement (EI) acquisition function in eqn (16) (used in Fig. 5) balances exploration and exploitation; acquiring the COF with the highest predicted deliverable capacity $\hat{y}$ is pure exploitation; acquiring the COF with the highest uncertainty σ in the predicted deliverable capacity is pure exploration (active learning). Fig. 7 shows the search efficiency of BO under these three different acquisition strategies, in terms of the highest deliverable capacity among the acquired set of COFs. Both the pure exploitation and pure exploration BO acquisition strategies exhibit subpar search performance compared to the EI acquisition strategy that trades off exploration and exploitation. The pure exploration acquisition strategy (active learning) performs the worst, as it acquires COFs with low deliverable capacities to reduce the

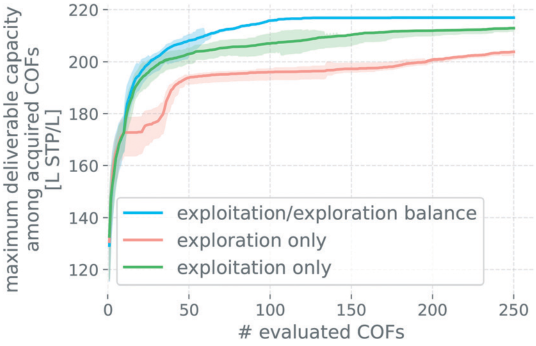


Fig. 7 BO search efficiency using three different acquisition functions: expected improvement (balances exploration and exploitation), the predicted deliverable capacity (full exploitation), and the uncertainty in the predicted deliverable capacity (full exploration).

uncertainty in the surrogate model's predictions about COFs with low deliverable capacities.

6.4 Feature importance

The search efficiency of BO is largely predicated on the accuracy of the surrogate model, which in turn is predicated on the information about the deliverable capacity of a COF provided by its feature vector. Here, we determine which COF features in Table 1 are most predictive of their deliverable capacity through a feature permutation importance study.

First, we randomly sample 5000 COFs (required owing to memory limitations with GPs) and split the 5000 (COF feature vector, deliverable capacity) pairs into an 80%/20% train/test set. Second, we fit a GP with the Matérn kernel (ν = 2.5) to the train set. The parity plot in Fig. 8a indicates the good performance of the trained GP on the test set (coefficient of determination R^2 = 0.9, root mean square error (RMSE) = 6.9). Third, we assess the importance of feature j for the predictions of the GP by randomly shuffling the values of feature j among the COFs in the test set and observing the deterioration of the R^2 of the GP on the test data set with this feature permuted (average over five shuffles). Fig. 8b displays the resulting feature importances. The two most important features are the crystal density and void fraction.

6.5 Conclusions from experiments

BO is an active search method to find the optimal NPM in a library while evaluating, with some expensive method such as a molecular simulation, only a small fraction of the NPMs in the data set. BO achieves this by leveraging a surrogate model to capture our beliefs about the structure–property relationship, given the observations thus far, to make principled acquisition decisions that balance exploration and exploitation. The adoption of BO could dramatically impact high-throughput computational screenings of NPMs by reducing the computing cost of finding the optimal NPM, allowing us to screen larger databases of NPMs, and enabling the use of higher-fidelity but more expensive molecular models and simulation methods. Notably, BO applies to NPM search in the experimental domain as well.

7 Outlook

We explained the key ideas behind Bayesian optimization (BO) and demonstrated its use to efficiently search databases of NPMs for the one with the optimal property, while synthesizing and evaluating the fewest NPMs. The ideas of BO, to sequentially, actively make intelligent decisions on which NPM to synthesize and evaluate based on the past experiments, can be applied to both the laboratory (driven by humans or robots[99–103]) and computational settings. The two core ingredients of BO are (1) a surrogate model that approximates the structure–property relationship and describes our uncertainty in it and (2) an acquisition function that scores the utility of evaluating each NPM next, designed to balance exploration and exploitation. We demonstrated BO

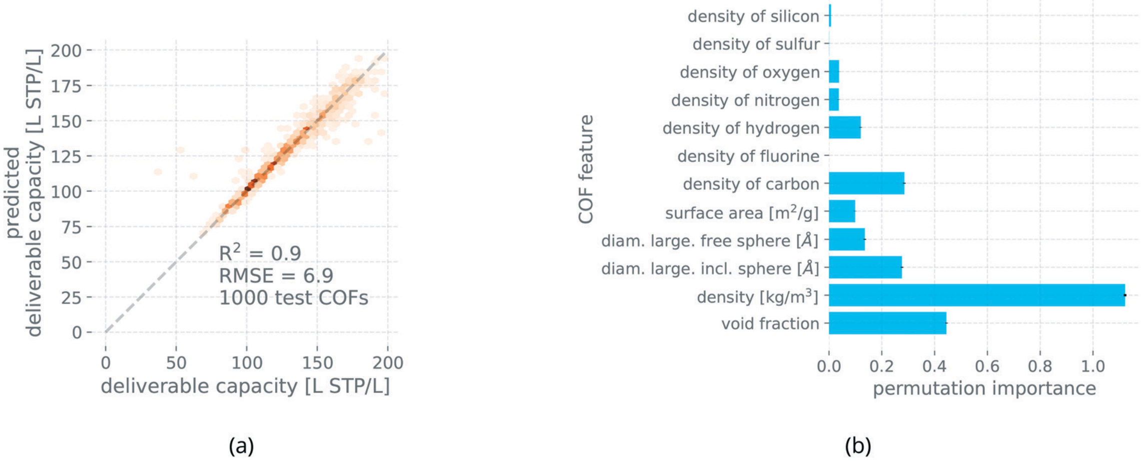


Fig. 8 Permutation feature importance in the GP model (using randomly selected set of 5000 COFs). (a) Parity plot showing the performance of the GP on the test set. (b) Permutation feature importances.

of NPMs by using BO to search through a database of *ca.* 70 000 COFs to find the COF with the highest simulated methane deliverable capacity; all 100 BO searches acquired the optimal COF after evaluating only 139 COFs. While preparing our article, Donval and Hand *et al.*[104] also demonstrated BO of MOFs and COFs for the acceleration of virtual screenings.

There are several extensions to and modifications of Bayesian optimization that are useful for different problem settings in NPM discovery:

- **Batch BO**

In standard BO, we select a single NPM to evaluate in each iteration. However, we may have parallel experimental resources to leverage to further accelerate the search for the optimal NPM. In *batch BO*,[97,105–110] at each BO iteration, we select multiple NPMs to synthesize and evaluate in parallel. Assuming the time required to evaluate each COF is the same, we expect batch BO to reduce the total time to find the optimal COF, compared to sequential BO. Assuming the resources required to evaluate each COF is the same, however, we expect batch BO to consume more resources to find the optimal COF, compared to sequential BO. The reason is that batch BO makes sub-optimal acquisition decisions for the second, third, and so forth acquired COF of each batch; *e.g.* the second decision would be better informed if we knew the outcome of the experiment from the first COF in the batch.

- **Multi-fidelity BO**

Often, we have a choice of multiple experimental methods to evaluate the property of an NPM. These methods usually involve a tradeoff in resource cost and the accuracy of the evaluation. For example, a molecular simulation of gas adsorption in an NPM is a low-fidelity experiment (cheap, but inaccurate) while measurement of gas adsorption in an NPM in a physical laboratory is a high-fidelity experiment (costly, but accurate). Intuitively, it is possible to leverage low-fidelity experiments to prune NPMs with low property values and to identify promising NPM candidates that can be searched further using high-fidelity experiments. In *multi-fidelity BO*,[111–120] we select both an NPM to evaluate and the fidelity of the experiment in each iteration. This allows optimization of the overall resource cost of experiments—of both low- and high-fidelity—for identifying high-performing NPMs.

- **Multi-objective BO**

We often need to optimize NPMs for multiple property objectives which are conflicting in nature and cannot be optimized simultaneously. For example, for gas separations, we often wish an NPM to have both a high selectivity and a high working capacity for the gas we wish to capture.[54] In [linear] scalarization, we aggregate the multiple objectives into a single objective by specifying a weight for each objective describing our priority for optimizing it.[121] However, such weights are often subjective or cannot be declared *a priori*. In such multi-objective optimization problems, we wish to find the Pareto optimal set of solutions and leave objective prioritization for downstream decision-makers who may weigh the objectives differently. A solution is Pareto optimal if it cannot be improved in any of the objectives without compromising some other objective. The goal of *multi-objective BO*[122–128] is to find the optimal Pareto set of NPMs using the fewest NPM evaluations. Similarly, the ε-PAL algorithm[129] has recently been used to find Pareto optimal polymers.

- **Constrained BO**

Possibly, some NPMs in the search space cannot be synthesized. More, often we cannot know if an NPM is synthesizable until we attempt its synthesis, which still incurs a cost. In this context, synthesizability is a black-box constraint over the search space. In *constrained BO*,[130–136] we perform BO where the synthesizability of an NPM cannot be verified without performing an experiment. The typical approach involves learning a statistical model based on the past evaluations of constraint(s) and selecting high-utility NPMs from the predicted feasible region (minimal to no constraint violation). Notably, a random forest classifier has been trained to predict the ease of synthesis of precursors for porous organic cages, using data collected from human experts on organic synthesis.[137] See ref. 77 for an overview of strategies to optimize molecules with synthesizability in consideration.

- **Cost-aware BO**

The evaluation cost can vary from one NPM to another (*e.g.*, cost of synthesizing NPM). We would like to take this cost into account to reduce the overall costs incurred during the search for the optimal NPM. In *cost-aware BO*,[138,139] the acquisition strategy considers not only the information gain of acquiring an NPM but also the cost incurred to synthesize it and measure its property.

- **Robust solutions to BO**

We may be uncertain about the measured/computed features of the NPMs and seek an optimum NPM that is robust to variations in its features. In *robust BO*, we account for the uncertainty in the inputs $\boldsymbol{x}$ when optimizing $f(\boldsymbol{x})$ and seek flat as opposed to sharp optima.[140,141]

- **BO that searches spaces with categorical or hybrid inputs**

In some cases, the search space could be composed of categorical as opposed to continuous variables, or a mix of them; developing BO frameworks that can handle categorical and hybrid search spaces is an active area of research.[142,143]

Some popular software packages for BO include BoTorch,[144] BayesOpt,[145] and SMAC.[146] COMBO[147] is a BO library tailored to materials science.

In addition to efficiently searching for NPMs with optimal properties, BO is applicable to a wide variety of optimization problems in the chemical and materials sciences.[148,149] BO has been used in both the laboratory and computational settings to efficiently search for optimal reaction conditions,[150–152] organic molecules,[153] ferroelectric materials,[154] compositions of and processing conditions for materials,[101,155] ligands to dock on proteins,[97,156,157] crystal structures,[110,158,159] shape memory alloys,[160] and density functional models.[161] For more general overview, see the reviews of Coley,[162] Coley *et al.*,[163] Terayama *et al.*,[164] Frazier and Wang,[165] and Lookman *et al.*[166]

The effectiveness of BO is predicted upon an accurate surrogate model of the structure–property relationship with calibrated uncertainty quantification.[167] In turn, the accuracy of the surrogate model is predicated on (i) an information-rich representation $\boldsymbol{x}$ of the NPM that encodes the salient features of its structure and chemical composition and (ii) a statistical model that (a) is sufficiently flexible/expressive to approximate the underlying objective function and (b) learns in a data-efficient manner. This gives important and currently active directions for future research. Particularly, engineering useful vector representations $\boldsymbol{x}$ of NPMs, using domain knowledge, is a very active research area. The representation should be invariant to rotations, translations, replications (if a crystal), and permutations of the list of atoms comprising the structure. Moreover, the mapping from NPM structures to feature vectors should be injective. Graph neural networks can, instead, *learn* vector representations of NPMs from their crystal structures represented as graphs with node labels. Recent work has developed graph neural networks capable of uncertainty quantification,[157,168] enabling the use of graph neural networks as surrogate models for BO.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

A. D. and J. R. D. acknowledge support from NSF grants IIS-1845922, OAC-1910213. C. M. S. acknowledges support from NSF grant 1920945. We thank the reviewers (incl. Jan Jensen) for helpful comments that led to improvements in our manuscript.

References

1 S. Ma and H.-C. Zhou, Gas storage in porous metal–organic frameworks for clean energy applications, *Chem. Commun.*, 2010, **46**(1), 44–53.

2 A. Schoedel, Z. Ji and O. M. Yaghi, The role of metal–organic frameworks in a carbon-neutral energy cycle, *Nat. Energy*, 2016, **1**(4), 16034.

3 B. Li, H.-M. Wen, W. Zhou and B. Chen, Porous metal–organic frameworks for gas storage and separation: What, how, and why?, *J. Phys. Chem. Lett.*, 2014, **5**(20), 3468–3479.

4 L. E. Kreno, K. Leong, O. K. Farha, M. Allendorf, R. P. Van Duyne and J. T. Hupp, Metal–organic framework materials as chemical sensors, *Chem. Rev.*, 2011, **112**(2), 1105–1125.

5 L. J. Murray, M. Dincă and J. R. Long, Hydrogen storage in metal–organic frameworks, *Chem. Soc. Rev.*, 2009, **38**(5), 1294.

6 K. Sumida, D. L. Rogow, J. A. Mason, T. M. McDonald, E. D. Bloch, Z. R. Herm, T.-H. Bae and J. R. Long, Carbon dioxide capture in metal–organic frameworks, *Chem. Rev.*, 2012, **112**(2), 724–781.

7 F.-Y. Yi, D. Chen, M.-K. Wu, L. Han and H.-L. Jiang, Chemical sensors based on metal–organic frameworks, *ChemPlusChem*, 2016, **81**(8), 675–690.

8 Z. Hu, B. J. Deibert and J. Li, Luminescent metal–organic frameworks for chemical sensing and explosive detection, *Chem. Soc. Rev.*, 2014, **43**(16), 5815–5840.

9 C. S. Diercks and O. M. Yaghi, The atom, the molecule, and the covalent organic framework, *Science*, 2017, **355**(6328), eaal1585.

10 X. Feng, X. Ding and D. Jiang, Covalent organic frameworks, *Chem. Soc. Rev.*, 2012, **41**(18), 6010.

11 M. S. Lohse and T. Bein, Covalent organic frameworks: Structures, synthesis, and applications, *Adv. Funct. Mater.*, 2018, **28**(33), 1705553.

12 S. Wan, F. Gándara, A. Asano, H. Furukawa, A. Saeki, S. K. Dey, L. Liao, M. W. Ambrogio, Y. Y. Botros, X. Duan, S. Seki, J. Fraser Stoddart and O. M. Yaghi, Covalent organic frameworks with high charge carrier mobility, *Chem. Mater.*, 2011, **23**(18), 4094–4097.

13 D. Ongari, A. V. Yakutovich, L. Talirz and B. Smit, Building a consistent and reproducible database for adsorption evaluation in covalent–organic frameworks, *ACS Cent. Sci.*, 2019, **5**(10), 1663–1675.

14 H. Furukawa, K. E. Cordova, M. O'Keeffe and O. M. Yaghi, The chemistry and applications of metal-organic frameworks, *Science*, 2013, **341**(6149), 1230444.

15 D. J. Tranchemontagne, Z. Ni, M. OKeeffe and O. M. Yaghi, Reticular chemistry of metal–organic polyhedra, *Angew. Chem., Int. Ed.*, 2008, **47**(28), 5136–5147.

16 T. Hasell and A. I. Cooper, Porous organic cages: soluble, modular and molecular pores, *Nat. Rev. Mater.*, 2016, **1**(9), 1–14.

17 M. Li, D. Li, M. O'Keeffe and O. M. Yaghi, Topological analysis of metal–organic frameworks with polytopic linkers and/or multiple building units and the minimal transitivity principle, *Chem. Rev.*, 2013, **114**(2), 1343–1370.

18 V. Santolini, M. Miklitz, E. Berardo and K. E. Jelfs, Topological landscapes of porous organic cages, *Nanoscale*, 2017, **9**(16), 5280–5298.

19 J. L. Segura, S. Royuela and M. Mar Ramos, Post-synthetic modification of covalent organic frameworks, *Chem. Soc. Rev.*, 2019, **48**(14), 3903–3945.

20 S. Mandal, S. Natarajan, P. Mani and A. Pankajakshan, Post-synthetic modification of metal–organic frameworks

toward applications, *Adv. Funct. Mater.*, 2020, **31**(4), 2006291.

21 P. G. Boyd, Y. Lee and B. Smit, Computational development of the nanoporous materials genome, *Nat. Rev. Mater.*, 2017, **2**(8), 17037.

22 Y. J. Colón, D. A. Gómez-Gualdrón and R. Q. Snurr, Topologically guided, automated construction of metal–organic frameworks and their evaluation for energy-related applications, *Cryst. Growth Des.*, 2017, **17**(11), 5801–5810.

23 L. Turcani, E. Berardo and K. E. Jelfs, stk : A python toolkit for supramolecular assembly, *J. Comput. Chem.*, 2018, **39**(23), 1931–1942.

24 P. G. Boyd and T. K. Woo, A generalized method for constructing hypothetical nanoporous materials of any net topology from graph theory, *CrystEngComm*, 2016, **18**(21), 3777–3792.

25 R. L. Martin and M. Haranczyk, Construction and characterization of structure models of crystalline porous polymers, *Cryst. Growth Des.*, 2014, **14**(5), 2431–2440.

26 A. Sturluson, M. T. Huynh, A. R. Kaija, C. Laird, S. Yoon, F. Hou, Z. Feng, C. E. Wilmer, Y. J. Colón, Y. G. Chung, D. W. Siderius and C. M. Simon, The role of molecular modelling and simulation in the discovery and deployment of metal-organic frameworks for gas storage and separation, *Mol. Simul.*, 2019, **45**(14–15), 1082–1121.

27 H. Daglar and S. Keskin, Recent advances, opportunities, and challenges in high-throughput computational screening of MOFs for gas separations, *Coord. Chem. Rev.*, 2020, **422**, 213470.

28 R. Mercado, R.-S. Fu, A. V. Yakutovich, L. Talirz, M. Haranczyk and B. Smit, In silico design of 2d and 3d covalent organic frameworks for methane storage applications, *Chem. Mater.*, 2018, **30**(15), 5069–5086.

29 K. M. Jablonka, D. Ongari, S. M. Moosavi and B. Smit, Big-data science in porous materials: Materials genomics and machine learning, *Chem. Rev.*, 2020, **120**(16), 8066–8129.

30 R. Pollice, G. dos Passos Gomes, M. Aldeghi, R. J. Hickman, M. Krenn, C. Lavigne, M. Lindner-D'Addario, A. K. Nigam, C. T. Ser, Z. Yao and A. Aspuru-Guzik, Data-driven strategies for accelerated materials design, *Acc. Chem. Res.*, 2021, **54**(4), 849–860.

31 S. Chong, S. Lee, B. Kim and J. Kim, Applications of machine learning in metal-organic frameworks, *Coord. Chem. Rev.*, 2020, **423**, 213–487.

32 S. Chibani and F.-X. Coudert, Machine learning approaches for the prediction of materials properties, *APL Mater.*, 2020, **8**(8), 080701.

33 S. M. Moosavi, K. M. Jablonka and B. Smit, The role of machine learning in the understanding and design of materials, *J. Am. Chem. Soc.*, 2020, **142**(48), 20273–20287.

34 K. Mukherjee and Y. J. Colón, Machine learning and descriptor selection for the computational discovery of metal-organic frameworks, *Mol. Simul.*, 2021, 1–21.

35 Z. Shi, W. Yang, X. Deng, C. Cai, Y. Yan, H. Liang, Z. Liu and Z. Qiao, Machine-learning-assisted high-throughput computational screening of high performance metal–organic frameworks, *Mol. Syst. Des. Eng.*, 2020, **5**(4), 725–742.

36 M. F. Langer, A. Goeßmann and M. Rupp, Representations of molecules and materials for interpolation of quantum-mechanical simulations via machine learning, 2020, arXiv preprint arXiv:2003.12081.

37 C. R. Collins, G. J. Gordon, O. A. Von Lilienfeld and D. J. Yaron, Constant size descriptors for accurate machine learning models of molecular properties, *J. Chem. Phys.*, 2018, **148**(24), 241718.

38 J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals and G. E. Dahl, Neural message passing for quantum chemistry, in *Proceedings of International Conference on Machine Learning (ICML)*, 2017, pp. 1263–1272.

39 H. Ohno and Y. Mukae, Machine learning approach for prediction and search: Application to methane storage in a metal–organic framework, *J. Phys. Chem. C*, 2016, **120**(42), 23963–23968.

40 C. M. Simon, R. Mercado, S. K. Schnell, B. Smit and M. Haranczyk, What are the best materials to separate a xenon/krypton mixture?, *Chem. Mater.*, 2015, **27**(12), 4459–4475.

41 S. M. Moosavi, A. Nandy, K. M. Jablonka, D. Ongari, J. P. Janet, P. G. Boyd, Y. Lee, B. Smit and H. J. Kulik, Understanding the diversity of the metal-organic framework ecosystem, *Nat. Commun.*, 2020, **11**(1), 4068.

42 E. H. Cho, X. Deng, C. Zou and L.-C. Lin, Machine learning-aided computational study of metal–organic frameworks for sour gas sweetening, *J. Phys. Chem. C*, 2020, 27580–27591.

43 J. Burner, L. Schwiedrzik, M. Krykunov, J. Luo, P. G. Boyd and T. K. Woo, High-performing deep learning regression models for predicting low-pressure CO2 adsorption properties of metal–organic frameworks, *J. Phys. Chem. C*, 2020, **124**(51), 27996–28005.

44 B. J. Bucior, N. S. Bobbitt, T. Islamoglu, S. Goswami, A. Gopalan, T. Yildirim, O. K. Farha, N. Bagheri and R. Q. Snurr, Energy-based descriptors to rapidly predict hydrogen storage in metal–organic frameworks, *Mol. Syst. Des. Eng.*, 2019, **4**(1), 162–174.

45 R. Anderson, J. Rodgers, E. Argueta, A. Biong and D. A. Gómez-Gualdrón, Role of pore chemistry and topology in the CO_2 capture capabilities of MOFs: From molecular simulation to machine learning, *Chem. Mater.*, 2018, **30**(18), 6325–6337.

46 G. S. Fanourgakis, K. Gkagkas, E. Tylianakis and G. Froudakis, A generic machine learning algorithm for the prediction of gas adsorption in nanoporous materials, *J. Phys. Chem. C*, 2020, **124**(13), 7117–7126.

47 M. Pardakhti, E. Moharreri, D. Wanik, S. L. Suib and R. Srivastava, Machine learning using combined structural and chemical descriptors for prediction of methane adsorption performance of metal organic frameworks (MOFs), *ACS Comb. Sci.*, 2017, **19**(10), 640–645.

48 M. Fernandez and A. S. Barnard, Geometrical properties can predict CO_2 and n_2 adsorption performance of metal–

organic frameworks (MOFs) at low pressure, *ACS Comb. Sci.*, 2016, **18**(5), 243–252.

49 H. Dureckova, M. Krykunov, M. Z. Aghaji and T. K. Woo, Robust machine learning models for predicting high CO_2 working capacity and CO_2/H_2 selectivity of gas adsorption in metal organic frameworks for precombustion carbon capture, *J. Phys. Chem. C*, 2019, **123**(7), 4133–4139.

50 X. Zhang, J. Cui, K. Zhang, J. Wu and Y. Lee, Machine learning prediction on properties of nanoporous materials utilizing pore geometry barcodes, *J. Chem. Inf. Model.*, 2019, **59**(11), 4636–4644.

51 Z. Li, B. J. Bucior, H. Chen, M. Haranczyk, J. I. Siepmann and R. Q. Snurr, Machine learning using host/guest energy histograms to predict adsorption in metal–organic frameworks: Application to short alkanes and Xe/Kr mixtures, *J. Chem. Phys.*, 2021, **155**(1), 014701.

52 A. Ahmed and D. J. Siegel, Predicting hydrogen storage in mofs via machine learning, *Patterns*, 2021, **2**(7), 100291.

53 T. C. Le and D. A. Winkler, Discovery and optimization of materials using evolutionary approaches, *Chem. Rev.*, 2016, **116**(10), 6107–6132.

54 Y. G. Chung, D. A. Gómez-Gualdrón, P. Li, K. T. Leperi, P. Deria, H. Zhang, N. A. Vermeulen, J. F. Stoddart, F. You, J. T. Hupp, O. K. Farha and R. Q. Snurr, In silico discovery of metal-organic frameworks for precombustion CO_2 capture using a genetic algorithm, *Sci. Adv.*, 2016, **2**(10), e1600909.

55 S. P. Collins, T. D. Daff, S. S. Piotrkowski and T. K. Woo, Materials design by evolutionary optimization of functional groups in metal-organic frameworks, *Sci. Adv.*, 2016, **2**(11), e1600954.

56 S. M. Moosavi, A. Chidambaram, L. Talirz, M. Haranczyk, K. C. Stylianou and B. Smit, Capturing chemical intuition in synthesis of metal-organic frameworks, *Nat. Commun.*, 2019, **10**(1), 539.

57 N. Beauregard, M. Pardakhti and R. Srivastava, In silico evolution of high-performing metal organic frameworks for methane adsorption, *J. Chem. Inf. Model.*, 2021, 3232–3239.

58 C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis and S. Colton, A survey of monte carlo tree search methods, *IEEE Trans. Comput. Intell. AI Games*, 2012, **4**(1), 1–43.

59 X. Zhang, K. Zhang and Y. Lee, Machine learning enabled tailor-made design of application-specific metal–organic frameworks, *ACS Appl. Mater. Interfaces*, 2019, **12**(1), 734–743.

60 X. Zhang, K. Zhang, H. Yoo and Y. Lee, Machine learning-driven discovery of metal–organic frameworks for efficient CO_2 capture in humid condition, *ACS Sustainable Chem. Eng.*, 2021, 2872–2879.

61 D. A. Cohn, Z. Ghahramani and M. I. Jordan, Active learning with statistical models, *J. Artif. Intell. Res.*, 1996, **4**, 129–145.

62 E. N. Muratov, J. Bajorath, R. P. Sheridan, I. V. Tetko, D. Filimonov, V. Poroikov, T. I. Oprea, I. I. Baskin, A. Varnek, A. Roitberg, O. Isayev, S. Curtalolo, D. Fourches, Y. Cohen, A. Aspuru-Guzik, D. A. Winkler, D. Agrafiotis, A. Cherkasov and A. Tropsha, QSAR without borders, *Chem. Soc. Rev.*, 2020, **49**(11), 3525–3564.

63 T. Le, V. Chandana Epa, F. R. Burden and D. A. Winkler, Quantitative structure-property relationship modeling of diverse materials properties, *Chem. Rev.*, 2012, **112**(5), 2889–2919.

64 B. Shahriari, K. Swersky, Z. Wang, R. P. Adams and N. De Freitas, Taking the human out of the loop: A review of bayesian optimization, *Proc. IEEE*, 2015, **104**(1), 148–175.

65 A. Agnihotri and N. Batra, *Exploring bayesian optimization*, Distill, 2020. https://distill.pub/2020/bayesian-optimization.

66 A. P. Bartók, R. Kondor and G. Csányi, On representing chemical environments, *Phys. Rev. B: Condens. Matter Mater. Phys.*, 2013, **87**(18), 184115.

67 L. Sarkisov, R. Bueno-Perez, M. Sutharson and D. Fairen-Jimenez, Materials informatics with PoreBlazer v4.0 and the CSD MOF database, *Chem. Mater.*, 2020, **32**(23), 9849–9867.

68 T. F. Willems, C. H. Rycroft, M. Kazi, J. C. Meza and M. Haranczyk, Algorithms and tools for high-throughput geometry-based analysis of crystalline porous materials, *Microporous Mesoporous Mater.*, 2012, **149**(1), 134–141.

69 L. Sarkisov and A. Harrison, Computational structure characterisation tools in application to ordered and disordered porous materials, *Mol. Simul.*, 2011, **37**(15), 1248–1257.

70 A. S. Rosen, S. M. Iyer, D. Ray, Z. Yao, A. Aspuru-Guzik, L. Gagliardi, J. M. Notestein and R. Q. Snurr, Machine learning the quantum-chemical properties of metal–organic frameworks for accelerated materials discovery, *Matter*, 2021, 1578–1597.

71 T. Xie and J. C. Grossman, Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties, *Phys. Rev. Lett.*, 2018, **120**(14), 145301.

72 A. Raza, F. Waqar, A. Sturluson, C. Simon and X. Fern, Towards explainable message passing networks for predicting carbon dioxide adsorption in metal-organic frameworks, 2020, arXiv preprint arXiv:2012.03723.

73 E. H. Cho and L.-C. Lin, Nanoporous material recognition via 3d convolutional neural networks: Prediction of adsorption properties, *J. Phys. Chem. Lett.*, 2021, **12**(9), 2279–2285.

74 A. Sturluson, M. T. Huynh, A. H. P. York and C. M. Simon, Eigencages: Learning a latent space of porous cage molecules, *ACS Cent. Sci.*, 2018, **4**(12), 1663–1676.

75 Y. LeCun, Y. Bengio and G. Hinton, Deep learning, *Nature*, 2015, **521**(7553), 436–444.

76 C. R. Collins, G. J. Gordon, O. A. von Lilienfeld and D. J. Yaron, Constant size descriptors for accurate machine learning models of molecular properties, *J. Chem. Phys.*, 2018, **148**(24), 241718.

77 Z. Yao, B. Sánchez-Lengeling, N. S. Bobbitt, B. J. Bucior, S. G. H. Kumar, S. P. Collins, T. Burns, T. K. Woo, O. K. Farha, R. Q. Snurr and A. Aspuru-Guzik, Inverse

design of nanoporous crystalline reticular materials with deep generative models, *Nat. Mach. Intell.*, 2021, **3**(1), 76–86.

78 M. Fernandez, N. R. Trefiak and T. K. Woo, Atomic property weighted radial distribution functions descriptors of metal–organic frameworks for the prediction of gas uptake capacity, *J. Phys. Chem. C*, 2013, **117**(27), 14095–14105.

79 Y. Lee, S. D. Barthel, P. Dłotko, S. M. Moosavi, K. Hess and B. Smit, High-throughput screening approach for nanoporous materials genome using topological data analysis: Application to zeolites, *J. Chem. Theory Comput.*, 2018, **14**(8), 4427–4437.

80 F. Faber, A. Lindmaa, O. A. von Lilienfeld and R. Armiento, Crystal structure representations for machine learning models of formation energies, *Int. J. Quantum Chem.*, 2015, **115**(16), 1094–1101.

81 UC Berkeley CS188 Intro to AI – Course Materials, http://ai.berkeley.edu/lecture_slides.html.

82 N. Srinivas, A. Krause, S. Kakade and M. Seeger, Gaussian process optimization in the bandit setting: No regret and experimental design, in *Proceedings of the 27th International Conference on on Machine Learning (ICML)*, 2010, pp. 1015–1022.

83 T. D. Bui, C. Nguyen and R. E. Turner, Streaming sparse Gaussian process approximations, in *Advances in Neural Information Processing Systems (NeurIPS)*, Curran Associates, Inc., 2017, vol. 30, pp. 3301–3309.

84 B. Settles, *Active Learning*, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers, 2012.

85 C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*, Adaptive computation and machine learning, MIT Press, 2006.

86 J. Görtler, R. Kehlbeck and O. Deussen, A visual exploration of gaussian processes, *Distill*, 2019, **4**(4), e17.

87 B. Schölkopf and A. J. Smola, *Learning with Kernels: support vector machines, regularization, optimization, and beyond*, Adaptive computation and machine learning series, MIT Press, 2002.

88 K. Borgwardt, E. Ghisu, F. Llinares-López, L. O'Bray and B. Rieck, Graph kernels: state-of-the-art and future challenges, 2020, arXiv preprint arXiv:2011.03854.

89 K. P. Murphy, *Machine learning: a probabilistic perspective*, MIT press, 2012.

90 F. Perronnin, J. Sánchez and T. Mensink, Improving the fisher kernel for large-scale image classification, in *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5–11, 2010, Proceedings, Part IV, Lecture Notes in Computer Science*, ed. K. Daniilidis, P. Maragos and N. Paragios, Springer, 2010, vol. 6314, pp. 143–156.

91 Z. Wang and S. Jegelka, Max-value entropy search for efficient bayesian optimization, in *International Conference on Machine Learning*, PMLR, 2017, pp. 3627–3635.

92 M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson and E. Bakshy, BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization, in *Advances in Neural Information Processing Systems 33*, 2020.

93 N. Hansen, The cma evolution strategy: a comparing review, *Towards a new evolutionary computation*, 2006, pp. 75–102.

94 N. Hansen, The cma evolution strategy: A tutorial, 2016, arXiv preprint arXiv:1604.00772.

95 D. C. Porumbel, J.-K. Hao and F. Glover, A simple and effective algorithm for the maxmin diversity problem, *Ann. Oper. Res.*, 2011, **186**(1), 275.

96 Q. P. Nguyen, S. Tay, B. K. H. Low and P. Jaillet, Top-k ranking Bayesian optimization, in *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, AAAI Press, 2021, pp. 9135–9143.

97 D. E. Graff, E. I. Shakhnovich and C. W. Coley, Accelerating high-throughput virtual screening through molecular pool-based active learning, *Chem. Sci.*, 2021, **12**(22), 7866–7881.

98 J. González, Introduction to bayesian optimization, *Gaussian process summer school at Sheffield University (slides)*, 2017.

99 R. L. Greenaway, V. Santolini, M. J. Bennison, B. M. Alston, C. J. Pugh, M. A. Little, M. Miklitz, E. G. B. Eden-Rump, R. Clowes, A. Shakil, H. J. Cuthbertson, H. Armstrong, M. E. Briggs, K. E. Jelfs and A. I. Cooper, High-throughput discovery of organic cages and catenanes using computational screening fused with robotic synthesis, *Nat. Commun.*, 2018, **9**(1), 2849.

100 C. W. Coley, D. A. Thomas, J. A. M. Lummiss, J. N. Jaworski, C. P. Breen, V. Schultz, T. Hart, J. S. Fishman, L. Rogers, H. Gao, R. W. Hicklin, P. P. Plehiers, J. Byington, J. S. Piotti, W. H. Green, A. J. Hart, T. F. Jamison and K. F. Jensen, A robotic platform for flow synthesis of organic compounds informed by AI planning, *Science*, 2019, **365**(6453), eaax1566.

101 B. P. MacLeod, F. G. L. Parlane, T. D. Morrissey, F. Häse, L. M. Roch, K. E. Dettelbach, R. Moreira, L. P. E. Yunker, M. B. Rooney, J. R. Deeth, V. Lai, G. J. Ng, H. Situ, R. H. Zhang, M. S. Elliott, T. H. Haley, D. J. Dvorak, A. Aspuru-Guzik, J. E. Hein and C. P. Berlinguette, Self-driving laboratory for accelerated discovery of thin-film materials, *Sci. Adv.*, 2020, **6**(20), eaaz8867.

102 R. L. Greenaway and K. E. Jelfs, Integrating computational and experimental workflows for accelerated organic materials discovery, *Adv. Mater.*, 2021, **33**(11), 2004831.

103 E. Stach, B. DeCost, A. G. Kusne, J. Hattrick-Simpers, K. A. Brown, K. G. Reyes, J. Schrier, S. Billinge, T. Buonassisi, I. Foster, C. P. Gomes, J. M. Gregoire, A. Mehta, J. Montoya, E. Olivetti, C. Park, E. Rotenberg, S. K. Saikin, S. Smullin, V. Stanev and B. Maruyama, Autonomous experimentation systems for materials development: A community perspective, *Matter*, 2021, 2702–2726.

104 G. Donval, C. Hand, J. Hook, E. Dupont, M. S. Landman, M. Freitag, M. Lennox and T. Düren, *Autonomous exploration and identification of high performing adsorbents using active learning*, 2021.

105 J. Azimi, A. Fern and X. Z. Fern, Batch Bayesian optimization via simulation matching, in *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6–9 December 2010, Vancouver, British Columbia, Canada*, ed. J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel and A. Culotta, Curran Associates, Inc., 2010, pp. 109–117.

106 T. Kathuria, A. Deshpande and P. Kohli, Batched Gaussian process bandit optimization via determinantal point processes, in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5–10, 2016, Barcelona, Spain*, ed. D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon and R. Garnett, 2016, pp. 4206–4214.

107 C. Angermüller, D. Belanger, A. Gane, Z. Mariet, D. Dohan, K. Murphy, L. Colwell and D. Sculley, Population-based black-box optimization for biological sequence design, in *Proceedings of the 37th International Conference on Machine Learning ICML, volume 119 of Proceedings of Machine Learning Research*, PMLR, 2020, pp. 324–334.

108 A. Deshwal, S. Belakaria and J. R. Doppa, Mercer features for efficient combinatorial bayesian optimization, in *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, AAAI Press, 2021, pp. 7210–7218.

109 F. Häse, L. M. Roch, C. Kreisbeck and A. Aspuru-Guzik, Phoenics: A Bayesian optimizer for chemistry, *ACS Cent. Sci.*, 2018, **4**(9), 1134–1145.

110 E. O. Pyzer-Knapp, L. Chen, G. M. Day and A. I. Cooper, Accelerating computational discovery of porous solids through improved navigation of energy-structure-function maps, *Sci. Adv.*, 2021, **7**(33), eabi4763.

111 R. Lam, D. L. Allaire and K. Willcox, Multifidelity optimization using statistical surrogate modeling for non-hierarchical information sources, in *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2015.

112 K. Kandasamy, G. Dasarathy, J. B. Oliva, J. Schneider and B. Poczós, Gaussian process bandit optimisation with multi-fidelity evaluations, in *Conference on Neural Information Processing Systems*, 2016.

113 Y. Zhang, T. N. Hoang, B. K. H. Low and M. Kankanhalli, Information-based multi-fidelity Bayesian optimization, in *Conference on Neural Information Processing Systems Workshop on Bayesian Optimization*, 2017.

114 J. Song, Y. Chen and Y. Yue, A general framework for multi-fidelity Bayesian optimization with Gaussian processes, *International Conference on Artificial Intelligence and Statistics*, 2019.

115 S. Takeno, H. Fukuoka, Y. Tsukada, T. Koyama, M. Shiga, I. Takeuchi and M. Karasuyama, Multi-fidelity Bayesian optimization with max-value entropy search, 2019, arXiv:1901.08275.

116 S. Belakaria, A. Deshwal and J. R. Doppa, Multi-fidelity multi-objective Bayesian optimization: An output space entropy search approach, in *AAAI conference on Artificial Intelligence (AAAI)*, 2020.

117 S. Belakaria, A. Deshwal and J. R. Doppa, *Information-theoretic multi-objective Bayesian optimization with continuous approximations*, CoRR, 2020, abs/2009.05700.

118 H. C. Herbol, P. Clancy and M. Poloczek, Cost-effective materials discovery: Bayesian optimization across information sources, *Mater. Horiz.*, 2020, **7**, 2113–2123.

119 O. Egorova, R. Hafizi, D. C. Woods and G. M. Day, Multifidelity statistical machine learning for molecular crystal structure prediction, *J. Phys. Chem. A*, 2020, **124**(39), 8065–8078, PMID: 32881496.

120 A. Tran, J. Tranchida, T. Wildey and A. P. Thompson, Multi-fidelity machine-learning with uncertainty quantification and bayesian optimization for materials design: Application to ternary random alloys, *J. Chem. Phys.*, 2020, **153**(7), 074705.

121 M. T. M. Emmerich and A. H. Deutz, A tutorial on multiobjective optimization: fundamentals and evolutionary methods, *Nat. Comput.*, 2018, **17**(3), 585–609.

122 D. Hernández-Lobato, J. Hernandez-Lobato, A. Shah and R. Adams, Predictive entropy search for multi-objective Bayesian optimization, in *ICML*, 2016, pp. 1492–1501.

123 S. Belakaria, A. Deshwal and J. R. Doppa, Max-value entropy search for multi-objective Bayesian optimization, in *NeurIPS*, 2019.

124 S. Belakaria, A. Deshwal, N. K. Jayakodi and J. R. Doppa, Uncertainty-aware search framework for multi-objective Bayesian optimization, in *AAAI*, 2020.

125 S. Suzuki, S. Takeno, T. Tamura, K. Shitara and M. Karasuyama, Multi-objective Bayesian optimization using pareto-frontier entropy, in *Proceedings of International Conference on Machine Learning (ICML)*, 2020, pp. 9279–9288.

126 A. M. Schweidtmann, A. D. Clayton, N. Holmes, E. Bradford, R. A. Bourne and A. A. Lapkin, Machine learning meets continuous flow chemistry: Automated optimization towards the pareto front of multiple objectives, *Chem. Eng. J.*, 2018, **352**, 277–282.

127 F. Häse, L. M. Roch and A. Aspuru-Guzik, Chimera: enabling hierarchy based multi-objective optimization for self-driving laboratories, *Chem. Sci.*, 2018, **9**(39), 7642–7655.

128 S. Belakaria, A. Deshwal and J. R. Doppa, Output space entropy search framework for multi-objective Bayesian optimization, *J. Artif. Intell. Res.*, 2021.

129 K. M. Jablonka, G. M. Jothiappan, S. Wang, B. Smit and B. Yoo, Bias free multiobjective active learning for materials design and discovery, *Nat. Commun.*, 2021, **12**(1), 1–10.

130 V. Perrone, I. Shcherbatyi, R. Jenatton, C. Archambeau and M. W. Seeger, *Constrained bayesian optimization with max-value entropy search*, CoRR, 2019, abs/1910.07003.

131 S. Belakaria, A. Deshwal and J. R. Doppa. *Max-value entropy search for multi-objective Bayesian optimization with constraints*, CoRR, 2020, abs/2009.01721.

132 S. Belakaria, A. Deshwal and J. R. Doppa. *Uncertainty aware search framework for multi-objective Bayesian optimization with constraints*, CoRR, 2020, abs/2008.07029.

133 Z. Zhou, S. Belakaria, A. Deshwal, W. Hong, J. R. Doppa, P. P. Pande and D. Heo, Design of multi-output switched-capacitor voltage regulator via machine learning, in *DATE*, 2020.

134 S. Belakaria, D. Jackson, Y. Cao, J. R. Doppa and X. Lu, Machine learning enabled fast multi-objective optimization for electrified aviation power system design, in *ECCE*, 2020.

135 R.-R. Griffiths and J. M. Hernández-Lobato, Constrained bayesian optimization for automatic chemical design using variational autoencoders, *Chem. Sci.*, 2020, **11**(2), 577–586.

136 M. A. Gelbart, J. Snoek and R. P. Adams, Bayesian optimization with unknown constraints, 2014, arXiv preprint arXiv:1403.5607.

137 S. Bennett, F. T. Szczypiński, L. Turcani, M. E. Briggs, R. L. Greenaway and K. E. Jelfs, Materials precursor score: Modeling chemists' intuition for the synthetic accessibility of porous organic cage precursors, *J. Chem. Inf. Model.*, 2021, 4342–4356.

138 E. H. Lee, V. Perrone, C. Archambeau and M. W. Seeger, *Cost-aware Bayesian optimization*, CoRR, 2020, abs/2003.10870.

139 G. Guinet, V. Perrone and C. Archambeau, *Pareto-efficient acquisition functions for cost-aware Bayesian optimization*, CoRR, 2020, abs/2011.11456.

140 M. Aldeghi, F. Häse, R. J. Hickman, I. Tamblyn and A. Aspuru-Guzik, Golem: An algorithm for robust experiment and process optimization, 2021, arXiv preprint arXiv:2103.03716.

141 L. Fröhlich, E. Klenske, J. Vinogradska, C. Daniel and M. Zeilinger, Noisy-input entropy search for efficient robust Bayesian optimization, in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2020, pp. 2262–2272.

142 F. Häse, L. M. Roch and A. Aspuru-Guzik, Gryffin: An algorithm for Bayesian optimization for categorical variables informed by physical intuition with applications to chemistry, 2020, arXiv preprint arXiv:2003.12127.

143 A. Deshwal, S. Belakaria and J. R. Doppa, Bayesian optimization over hybrid spaces, in *Proceedings of International Conference on Machine Learning (ICML)*, 2021.

144 M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson and E. Bakshy, BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization, in *Advances in Neural Information Processing Systems 33*, 2020.

145 R. Martinez-Cantin, BayesOpt: a Bayesian optimization library for nonlinear optimization, experimental design and bandits, *J Mach Learn Res*, 2014, **15**(1), 3735–3739.

146 F. Hutter, H. H. Hoos and K. Leyton-Brown, Sequential model-based optimization for general algorithm configuration, *Technical Report TR-2010-10*, University of British Columbia, Department of Computer Science, 2010.

147 T. Ueno, T. D. Rhone, Z. Hou, T. Mizoguchi and Koji Tsuda, COMBO: An efficient Bayesian optimization library for materials science, *Mater. Discov.*, 2016, **4**, 18–21.

148 Q. Liang, A. E. Gongora, Z. Ren, A. Tiihonen, Z. Liu, S. Sun, J. R. Deneault, D. Bash, F. Mekki-Berrada, S. A. Khan, K. Hippalgaonkar, B. Maruyama, K. A. Brown, J. Fisher III and T. Buonassisi, Benchmarking the performance of Bayesian optimization across multiple experimental materials science domains, 2021, arXiv preprint arXiv:2106.01309.

149 J. R. Doppa, Adaptive experimental design for optimizing combinatorial structures, in *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

150 B. J. Shields, J. Stevens, J. Li, M. Parasram, F. Damani, J. I. Martinez Alvarado, J. M. Janey, R. P. Adams and A. G. Doyle, Bayesian reaction optimization as a tool for chemical synthesis, *Nature*, 2021, **590**(7844), 89–96.

151 A. M. Schweidtmann, A. D. Clayton, N. Holmes, E. Bradford, R. A. Bourne and A. A. Lapkin, Machine learning meets continuous flow chemistry: Automated optimization towards the pareto front of multiple objectives, *Chem. Eng. J.*, 2018, **352**, 277–282.

152 M. Christensen, L. P. E. Yunker, F. Adedeji, F. Häse, L. M. Roch, T. Gensch, G. dos Passos Gomes, T. Zepel, M. S. Sigman, A. Aspuru-Guzik and J. E. Hein, Data-science driven autonomous process optimization, *Commun. Chem.*, 2021, **4**(1), 112.

153 K. Korovina, S. Xu, K. Kandasamy, W. Neiswanger, B. Poczos, J. Schneider and E. Xing, Chembo: Bayesian optimization of small organic molecules with synthesizable recommendations, in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2020, pp. 3393–3403.

154 A. Biswas, A. N. Morozovska, M. Ziatdinov, E. A. Eliseev and S. V. Kalinin, *Multi-objective bayesian optimization of ferroelectric materials with interfacial control for memory and energy storage applications*, 2021.

155 B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes, N. Rankin, B. Harris, R. S. Sprick and A. I. Cooper, A mobile robotic chemist, *Nature*, 2020, **583**(7815), 237–241.

156 S. Mehta, S. Laghuvarapu, Y. Pathak, A. Sethi, M. Alvala and U. D. D. Priyakumar, Memes: Machine learning framework for enhanced molecular screening, *Chem. Sci.*, 2021, 11710–11721.

157 A. P. Soleimany, A. Amini, S. Goldman, D. Rus, S. N. Bhatia and C. W. Coley, Evidential deep learning for guided molecular property prediction and discovery, *ACS Cent. Sci.*, 2021, 1356–1367.

158 T. Yamashita, N. Sato, H. Kino, T. Miyake, K. Tsuda and T. Oguchi, Crystal structure prediction accelerated by Bayesian optimization, *Phys. Rev. Mater.*, 2018, **2**(1), 013803.

159 E. Pyzer-Knapp, G. Day, L. Chen and A. I. Cooper, Distributed multi-objective Bayesian optimization for the intelligent navigation of energy structure function maps for efficient property discovery, *ChemRxiv*, 2020.

160 D. Xue, P. V. Balachandran, J. Hogden, J. Theiler, D. Xue and T. Lookman, Accelerated search for materials with targeted properties by adaptive design, *Nat. Commun.*, 2016, **7**(1), 11241.

161 R. A. Vargas-Hernández, Bayesian optimization for calibrating and selecting hybrid-density functional models, *J. Phys. Chem. A*, 2020, **124**(20), 4053–4061.

162 C. W. Coley, Defining and exploring chemical spaces, *Trends Chem.*, 2021, **3**(2), 133–145.

163 C. W. Coley, N. S. Eyke and K. F. Jensen, Autonomous discovery in the chemical sciences part i: Progress, *Angew. Chem., Int. Ed.*, 2020, **59**(51), 22858–22893.

164 K. Terayama, M. Sumita, R. Tamura and K. Tsuda, Black-box optimization for automated discovery, *Acc. Chem. Res.*, 2021, **54**(6), 1334–1346.

165 P. I. Frazier and J. Wang, Bayesian optimization for materials design, in *Information Science for Materials Discovery and Design*, Springer, 2016, pp. 45–75.

166 T. Lookman, P. V. Balachandran, D. Xue and R. Yuan, Active learning in materials science with emphasis on adaptive sampling using uncertainties for targeted design, *npj Comput. Mater.*, 2019, **5**(1), 1–17.

167 K. Tran, W. Neiswanger, J. Yoon, Q. Zhang, E. Xing and Z. W. Ulissi, Methods for comparing uncertainty quantifications for material property predictions, *Mach. Learn.: Sci. Technol.*, 2020, **1**(2), 025006.

168 J. Allotey, K. T. Butler and J. Thiyagalingam, Entropy-based active learning of graph neural network surrogate models for materials properties, 2021, arXiv preprint arXiv:2108.02077.

169 J. M. Hernandez-Loba, M. W. Hoffman and Z. Ghahramani, Predictive Entropy Search for Efficient Global Optimization of Black-box Functions, *NIPS Workshop on Bayesian Optimization in Academia and Industry*, 2014.

170 K. Mukherjee, A. W. Dowling and Y. J. Colon, Sequential Design of Adsorption Simulations in Metal-Organic Frameworks, 2021, arXiv, arXiv:2110.00069v1.