Received: 9 August 2021 | Revised: 5 January 2022 | Accepted: 15 January 2022

DOI: 10.1002/aic.17616

PROCESS SYSTEMS ENGINEERING

AIChE JOURNAL

Efficient surrogates construction of chemical processes: Case studies on pressure swing adsorption and gas-to-liquids

Zhimian Hao[1] | Chonghuan Zhang[1] | Alexei A. Lapkin[1,2]

[1]Department of Chemical Engineering and Biotechnology, University of Cambridge, Cambridge, UK

[2]Cambridge Centre for Advanced Research and Education in Singapore, CARES Ltd, Singapore

Correspondence
Alexei A. Lapkin, Department of Chemical Engineering and Biotechnology, University of Cambridge, Cambridge CB3 0AS, UK.
Email: aal35@cam.ac.uk

Funding information
Chinese Scholarship Council; Cambridge Trust; National Research Foundation Singapore, CREATE: CARES, C4T Project

Abstract

We propose a sequential sampling approach to training statistical digital twins. This approach is relevant for real-world engineering problems with expensive data generation. Prerequisite for building surrogates is sufficient data; however, oversampling does not improve regression accuracy. The time for data generation may be reduced by: (a) applying a classifier to improve data quality and avoid evaluation of infeasible inputs, and (b) employing dynamic sampling linked to regression quality. In dynamic sampling, the initial sampling rate is large to generate enough data for surrogate regression in a few iterations; the sampling rate gradually slows down with the improvement of the iteratively refined surrogate. A dynamic process and a steady-state process from the field of carbon capture and utilization are used as case studies: pressure swing adsorption (PSA) and gas-to-liquids (GTL). The computational costs for surrogates generation are reduced by 86% for PSA and 51% for GTL, compared with employing a static sampling rate.

KEYWORDS
data quality, digital twins, dynamic sampling rate, gas-to-liquids, machine learning, pressure swing adsorption

1 | INTRODUCTION

The transformation to Industry 4.0 is driven by the advancements in digitalization.[1] As a foundation of digitalization, digital twins, referring to surrogates in this work, can represent the physical assets within cyber domain, and play an important role in the evaluation of engineering systems.[2] To build surrogates, one of prerequisites is data, the generation of which can be time-consuming and prohibitively expensive for real-world engineering systems. Conventional sampling methods can lead to under/oversampling issues.[3] Our strategy is to develop a workflow to reduce the total time spent on data generation by: (a) lowering the total number of the required data points, and (b) shortening the time per quantum of data generation. To successfully set up such a methodology, it is beneficial to review prior works on surrogate modeling and sampling methods, which will be elaborated in the remainder of this section.

The evaluation of real-world engineering systems is expensive. With physical models and inputs, computer simulations can accurately deliver information about systems. Although cheaper than experimental or industrial data, simulations can be considerably slow when the systems involve multiscale, multiphase phenomena, and dynamic behaviors.[4] Data-driven surrogates can represent the original physical models by building relationship between inputs and outputs (responses). For some complex systems, there are even no physical models available, and thus surrogates together with the design of experiments (DoEs) seem to be the only choice.[5–7] Surrogates are cheap-to-evaluate and can directly be employed for optimization, control, and design in many engineering fields, for example, chemical

engineering,[8–11] pharmaceutical manufacturing,[12] supply chain management,[13,14] and aerospace engineering.[15–17]

Besides the conventional formulations of surrogates, the booming of machine learning has expanded the "surrogate family"[4,9,10,18] with more choices, for example, artificial neural networks (ANNs) and Gaussian process (GP).[19–22] ANNs are regarded as universal approximators[23] and allow to fit multiple output variables simultaneously. The flexible structures and various activation functions enable ANNs to accurately fit any linear or nonlinear relationship of input/output. GP, often referred as Kriging, belongs to a nonparametric model type, which has excellent fitting performance and can predict uncertainty.[20] However, GP is generally implemented for single output variable, whereas the formulation for fitting multiple output variables is complex.[24,25]

Data is one of the prerequisites to train a surrogate. When the available data is limited, data generation is necessary, but it can be extremely expensive for real-world engineering systems.[19] To demonstrate it, we consider an example of a chemical process – pressure swing adsorption (PSA).[26–28] Details about PSA can be found in Case study 1 (Section 4.1). Data is randomly sampled, while the surrogate is trained iteratively. As shown in Figure 1, computational cost for data generation has a significantly higher order of magnitude than that for surrogate training, which is one of the common problems in chemical process systems. Insufficient data quantity cannot guarantee good quality for constructing a surrogate, while Garud et al. review that simply increasing the data quantity cannot lead to better performance of a surrogate.[3] Thus, the quality of surrogates should rely on both data quantity *and* quality.

To reduce the time for data generation, the first objective is to obtain good-enough surrogates with the minimum amount of data. There are two types of sampling methods: one-shot and sequential (adaptive) methods.[10] The former method samples the design space uniformly in one go and then builds a surrogate, while the latter samples data in batch and refine the surrogate iteratively. The one-shot is straightforward but may result in under/oversampling, where either poor regression or inefficiency can occur. In recent years, the sequential methods tend to be popular because they are reported to better balance the regression performance and efficiency.[10]

However, oversampling is still hard to avoid by a typical sequential method. To demonstrate this, we still use the example of a PSA process. Mean squared errors (MSEs) are employed to evaluate the regression performance. The data is sequentially sampled by Latin hypercube sampling (LHS), which is stochastic. A stochastic method has an anytime behavior,[29] where sampling can be stopped at any time. We plot the fitting performance against time to observe the termination condition as shown in Figure 2. We divide the sequential sampling as two equal parts based on the number of data points (approximately equivalent to time, because the time for surrogate training can be neglected compared with data generation as shown in Figure 1). The plot indicates that regression improvement in the first half is significantly greater than that in the second half. This suggests that too much data is not worth collecting, although the limit of the infinite number of data points is required to fully fill the searching space theoretically. In other words, due to the anytime behavior, we should stop any further sampling after achieving a certain fitting performance. Also, we noticed that the MSE values fluctuate all the time. Hence, it is rather challenging to determine an optimal termination criterion.

Meanwhile, with the linear increase in the number of data points generated, time seems to exponentially increase for surrogate training (Figure 1). Consequently, the surrogate training might be extremely time-consuming if the number of sampled data points is high. Therefore, oversampling brings the unnecessary time costs for data generation and extra effort for surrogate training. To avoid oversampling, it might be beneficial to spot the nonimprovement trend as early as possible.

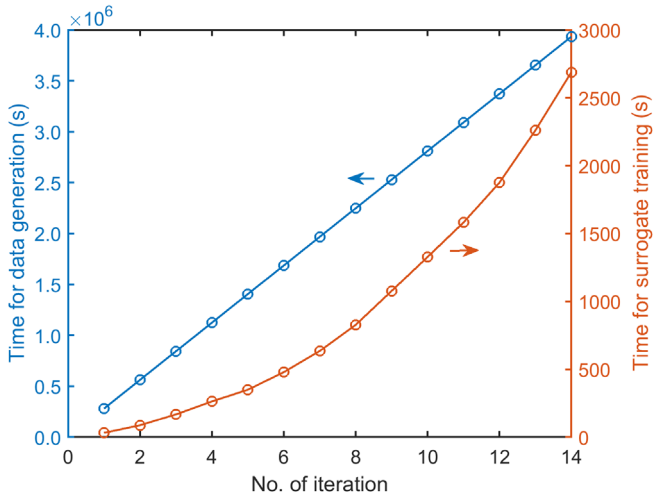

FIGURE 1 Computational cost on data generation vs. surrogate training for the PSA process

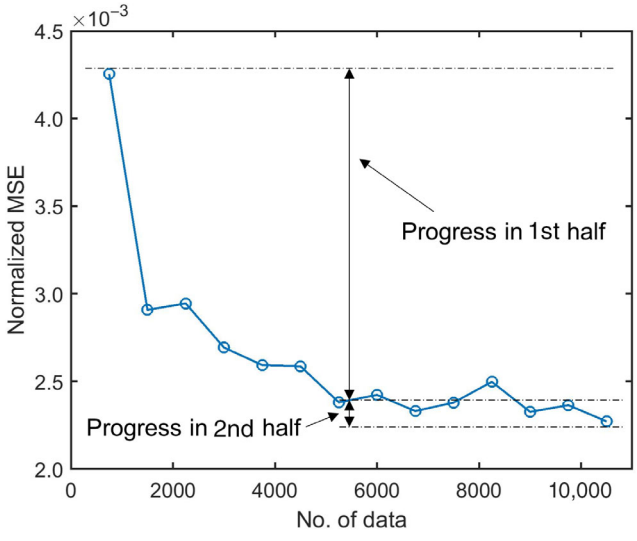

FIGURE 2 Illustration of why too many data points might not be worth sampling

The second objective in sampling is concerned with the improvement of data quality. The design space for sampling is initially based on limited prior experience or even random guesses, and the infeasible design space is commonly unavoidable. Consequently, some inputs, which happen to be sampled from the infeasible design space, can lead to unexpected outputs, such as nonconverged simulation outputs or experimental failures. Such outputs will introduce significant errors to the surrogate construction, and thus they are ineffective data, which should be screened out. To increase data effectiveness, a classifier can be constructed to distinguish between infeasible and feasible design spaces. Such application of a classifier has been successfully demonstrated in prior research works. Ibrahim et al. reported that a support vector machine (SVM) can be used to set a feasibility constraint to filter infeasible design space for nonconverged simulations.[30] Cao et al. adopted a Bayes classifier to improve the design space for the experimental conditions of formulations.[5] Kim et al. applied a combined classification system to increase the quality of design space for computation-based material discovery, which can significantly reduce the number of further samplings.[31] Houben et al. included a classifier into a Bayesian optimization algorithm to avoid infeasible experiments in emulsion polymerization.[32]

To further enhance data quality, exploitation-based methods can be considered to identify the promising sample placement. A simple example can be used to demonstrate the importance of sample placement. Suppose a data-driven model fits a model with a simple form $y = \sin(x)$ at the design space $[0,\pi]$. Then the sampling places at $0, \frac{\pi}{2}, \pi$ are more important than other places. The real-world engineering problems can be more complex with high dimensionality, and exploitation-based methods tend to place more samples in the highly nonlinear/complex regions.[10] Cozad et al. develop a workflow called ALAMO for algebraic model building in a sequential sampling way.[33,34] For a new data to sample, they apply a derivative-free optimization technique to identify the sample placement, which holds the largest error between the surrogate and the original model. To identify one optimal sample placement, many new data points are required to be generated for evaluation during the optimization. Consequently, this method, actually, generates far more data points than the reported number of optimal data points. An alternative approach is to employ GP-based surrogates, which can predict the model errors. The region with the maximum prediction error is selected for new points.[35,36] However, this approach is limited to GP-based surrogate type, since the error prediction is not a generic characteristic for other surrogate types.[3] Garud et al. review that the surrogate-independent strategies can be more advantageous, because they can be more generic and can guarantee sampling randomness.[3] Most of these strategies are based on certain score criteria to identify complex regions, which then require exploitation-based methods for local improvement. Since it is out of the scope of this work, more detailed information can be referred to in the Garud's review article.[3] Although the exploitation-based approaches are powerful in improving data quality, the complex mathematical formulations make them difficult in implementation.

In this work we aim to develop a generic and easy-to-implement sampling method for surrogate generation. The sampling efficiency benefits from:

- reduction in the total number of sampling points;
- reduction in the time per data generation.

The remainder of this work is structured as follows: Section 2 proposes the overall workflow for the surrogate construction; Section 3 demonstrates the state-of-art of two principles for efficient data generation; further, Section 4 presents two case studies on chemical processes, followed by conclusions and outlook in the final section.

2 | WORKFLOW FOR SURROGATE CONSTRUCTION

This section presents the workflow for surrogate generation. We select LHS as the sampling technique, because it does not lose generality with the increase of dimensionality and can deliver a well-distributed sampling result.[3] As shown in Figure 3, the algorithm samples initial data by LHS. Then, simulations or experiments generate the corresponding outputs. With the initial data points (or together with a few iterations), an SVM classifier is trained to separate the feasible design space from the infeasible one. The data inputs from the infeasible region are deleted, while inputs in the feasible region are passed to the simulator for outputs. To fit multiple outputs simultaneously, ANN is selected as the surrogate type. In the successive iterations, data is sampled in batch by LHS for surrogate refinement, with which the sampling rate gradually slows down.

We briefly demonstrate the procedure for data generation. Process simulators—Aspen Plus, Dymola, or gPROMS are powerful tools for process modeling. Still, they are not flexible for data storage and are limited in their capacity to access high-level statistic packages. Therefore, it is necessary to establish an interface between process simulators and high-level programming languages (e.g., MATLAB, Python, etc.). Advantages of MATLAB over other programming languages are: reliable optimization and machine-learning toolboxes, which have been well-established for commercial use. The automation of MATLAB to process simulators has previously been reported in process design[37–39] and control.[40] As shown in Figure 4, the inputs are sampled by LHS in MATLAB and passed to simulators, for example, Dymola for dynamic simulations and Aspen Plus for steady-state process simulations. The obtained outputs are sent back to MATLAB for data collection.

At each sequential sampling iteration (*i*th iteration), the workflow can generate a surrogate (Surrogate_i). The regression performance of Surrogate_i is computed by a training–validation–test method. Specifically, the obtained dataset is divided into three subsets: training, validation, and test at a ratio 70%/20%/10%. Given the nonlinearity of process systems, we use a nonlinear activation function—hyperbolic

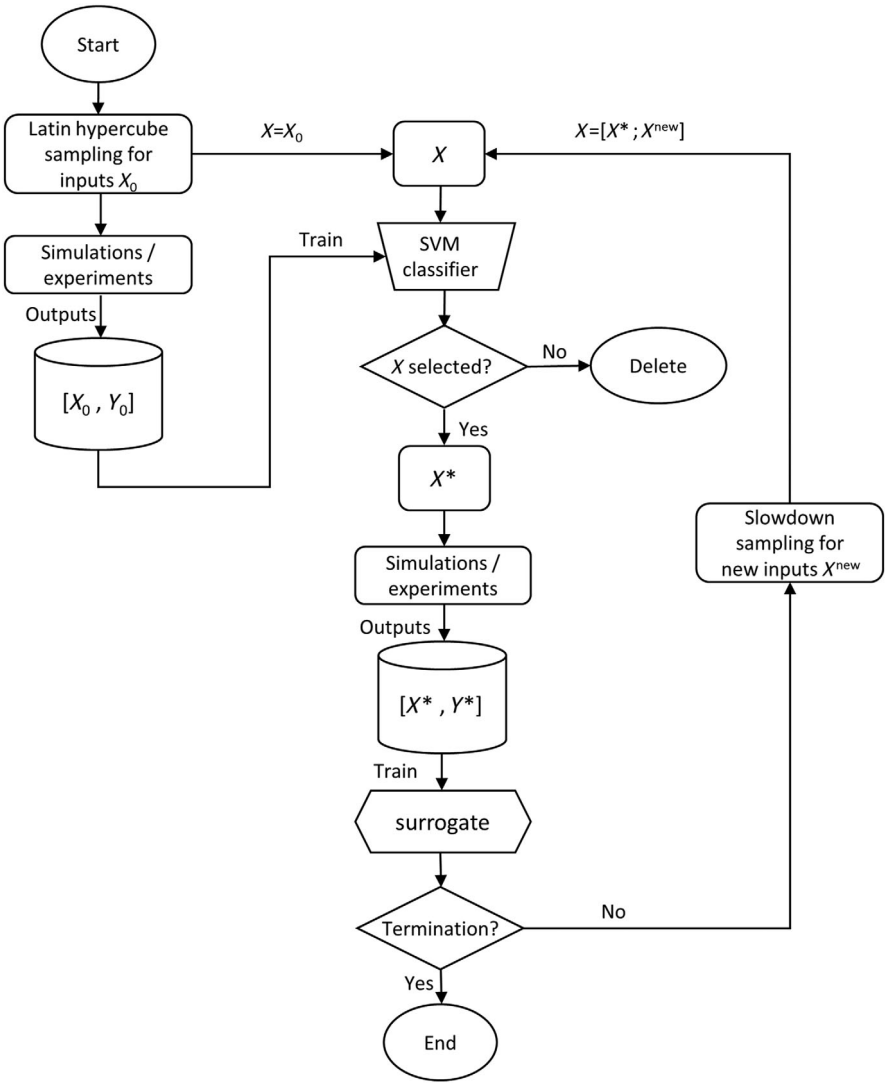

FIGURE 3 Proposed workflow for surrogate generation. $[X_0, Y_0]$ are initial inputs/outputs to train an SVM classifier; $[X^*, Y^*]$ are the inputs/outputs (selected by SVM classifier) for surrogate training in the latest iteration; X^{new} are the inputs for the next iteration; X are the updated inputs in the latest iteration. The added number of samples in iteration i refers to the sampling rate (N_{added_i}) in iteration i

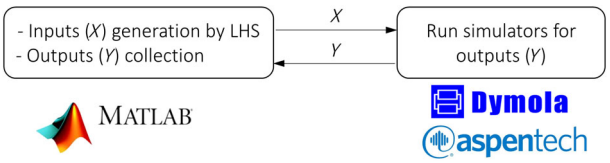

FIGURE 4 Data generation by interfacing MATLAB with process simulators

tangent (tanh). We optimize the structure of networks by a random search strategy. In the random search strategy, a set of network candidates are established with random structures (e.g., the number of layers and the number of neurons is different within the network candidates), and they are regressed by the training dataset. Following this, these trained network candidates are evaluated using validation dataset, and the network candidate with the minimal MSE value is selected as Surrogate_i. The regression performance of Surrogate_i is determined by its MSE based on the test dataset.

3 | STATE-OF-ART FOR EFFICIENT DATA GENERATION

The two principles, the classifier and slowdown sampling, are detailed in this section.

3.1 | Classifier SVM

The sampled points might fall in the infeasible design space due to extreme operating conditions for experiments (e.g., unexpected reactions occur at high temperature) or nonconverged recycle streams, or integration failure on stiff models during computational simulations. A classifier can be trained to pretreat the data inputs. Only the selected data inputs can be passed into the simulation or experiment stage, thus saving the average time spent on a single data point.

SVM is a machine learning technique primarily for classification. SVM was initially proposed as a linear classifier, while Vapnik et al.

expanded its application as a nonlinear classifier in 1995.[41] It is a mature and reliable method, the success of which was proven in the fields of pattern recognition and computer vision problems.[42] For a typical chemical process, the high-dimensional features (or multiple inputs) and the nonlinearity are unavoidable. Ibrahim has demonstrated its successful application in chemical process engineering.[30] A toolbox of SVM can be accessed in MATLAB, so SVM is selected as the classifier in this work.

The training process for SVM is similar to the steps for surrogate training. Two differences are specified here. Firstly, only the dataset in the initial several iterations is used to train the classifier. This is because the classifier in this work is expected to give rough classification between infeasible and feasible design spaces, so the iterative refinement for the classifier is not necessary. Secondly, the output for the classifier is binary, 0 and 1: set 0 if the simulation outputs fall on the infeasible space, while setting 1 if the simulation outputs fall on the feasible area. Following this, the data inputs together with the classifier outputs are used to train the SVM.

3.2 | Slowdown sampling

To clearly explain the slowdown sampling strategy, we start with the definition of two variables as follows:

- Sampling rate (N_{added_i}): the number of new samples in ith iteration.
- Surrogate improvement rate ($\left|\text{slope}_{\overline{\text{MSE}_i}}\right|$): the surrogate improvement per sample added.

3.2.1 | Logic behind slowdown sampling

When employing sequential sampling based on a static sampling rate, a practical question falls on how to determine a proper value for the sampling rate. A large rate can result in the oversampling in the final iterations, while a small rate will lead to too many iterations, but the training in the early iterations is not meaningful based on a small dataset. Herein, we propose that the sampling rate can be dynamic: initially the sampling rate is relatively large as to achieve a reasonable data quantity for surrogate regression in just a few iterations; the sampling rate gradually slows down with the regression improvement of the iteratively refined surrogate. This refers to the slowdown sampling principle. To achieve this, we need to build the relationship between sampling rate and surrogate improvement rate.

First, we explain how to quantify the surrogate improvement rate ($\left|\text{slope}_{\overline{\text{MSE}_i}}\right|$) in ith iteration. The first iteration obtains the result directly from the classifier section. For a successive iteration ($i \geq 2$), MSE_i is used to quantify the regression performance. We use the moving mean ($\overline{\text{MSE}_i}$) to smooth the fluctuation of the MSE curve. The MSE decrease per data added, or we call it the $\text{slope}_{\overline{\text{MSE}_i}}$, is defined as Equation (1). Its absolute value can reflect on how the surrogate can be refined based on one more data, so $\left|\text{slope}_{\overline{\text{MSE}_i}}\right|$ is suitable to express the surrogate improvement rate.

$$\text{slope}_{\overline{\text{MSE}_i}} = \frac{\overline{\text{MSE}_i} - \overline{\text{MSE}_{i-1}}}{N_{\text{added}_i}} \tag{1}$$

Second, we propose how the sampling rate is expected to respond to the surrogate improvement rate. A large value of $\left|\text{slope}_{\overline{\text{MSE}_i}}\right|$ indicates that addition of new samples can significantly improve the quality of the surrogate; hence, sampling rate of the next iteration ($N_{\text{added}_{i+1}}$) is expected to be large; while a very small value of $\left|\text{slope}_{\overline{\text{MSE}_i}}\right|$ indicates that oversampling tends to occur, so $N_{\text{added}_{i+1}}$ should approach 0. In brief, the smaller $\left|\text{slope}_{\overline{\text{MSE}_i}}\right|$ is, the smaller $N_{\text{added}_{i+1}}$ is.

Third, we display the steps of relating the surrogate improvement rate ($\left|\text{slope}_{\overline{\text{MSE}_i}}\right|$) to the sampling rate ($N_{\text{added}_{i+1}}$).

- Step 1: As Equation (2), $\text{slope}_{\overline{\text{MSE}_i}}$ can be scaled to a relative slope ($\text{slope}_{\text{relative}_i}$), based on the scale—initial slope value ($\text{slope}_{\text{MSE}_2}$). In most cases, the surrogate improvement rate is largest in the beginning, so the $\text{slope}_{\text{MSE}_2}$ normally has the largest absolute value among all the $\text{slope}_{\overline{\text{MSE}_i}}$. Thus, the relative slope value normally falls between -1 and 1 (due to the fluctuation, the value of the slope can be positive). The absolute value of relative slope ($\left|\text{slope}_{\text{relative}_i}\right|$) reflects how $\left|\text{slope}_{\overline{\text{MSE}_i}}\right|$ drops, when comparing to the initial surrogate improvement rate.

$$\text{slope}_{\text{relative}_i} = \frac{\text{slope}_{\overline{\text{MSE}_i}}}{\text{slope}_{\text{MSE}_2}} \tag{2}$$

- Step 2: A ratio function can convert the relative slope to a positive value as the added ratio ($\text{added}_{\text{ratio}_{i+1}}$, typically between 0 and 1). $\text{added}_{\text{ratio}_{i+1}}$ refers to ratio of the sampling rate over the maximum sampling rate. The formula for the ratio function can be found at Section S1 in Supporting Information. Step 2 aims to achieve "the smaller $\left|\text{slope}_{\text{MSE}_i}\right|$ is, the smaller $N_{\text{added}_{i+1}}$ is."

$$\text{added}_{\text{ratio}_{i+1}} = \text{ratio}_{\text{function}}\left(\left|\text{slope}_{\text{relative}_i}\right|\right) \tag{3}$$

- Step 3: The sampling rate ($N_{\text{added}_{i+1}}$) is calculated through multiplying $\text{added}_{\text{ratio}_i}$ by the maximum sampling rate (the maximum number of new samples per iteration, N_{upper}), see Equation (4).

$$N_{\text{added}_{i+1}} = N_{\text{upper}} \times \text{added}_{\text{ratio}_{i+1}} \tag{4}$$

3.2.2 | Demonstration of slowdown sampling by fitting kinetics for $A \xrightarrow{k_1} B \xrightarrow{k_2} C$

To better demonstrate the slowdown sampling, we use a simple example of fitting two reactions in series $A \xrightarrow{k_1} B \xrightarrow{k_2} C$. The two reactions are assumed to obey first-order kinetics, as written in Equations (5)–(7).

$$\frac{dA}{dt} = -k_1 A \tag{5}$$

$$\frac{dB}{dt} = k_1 A - k_2 B \tag{6}$$

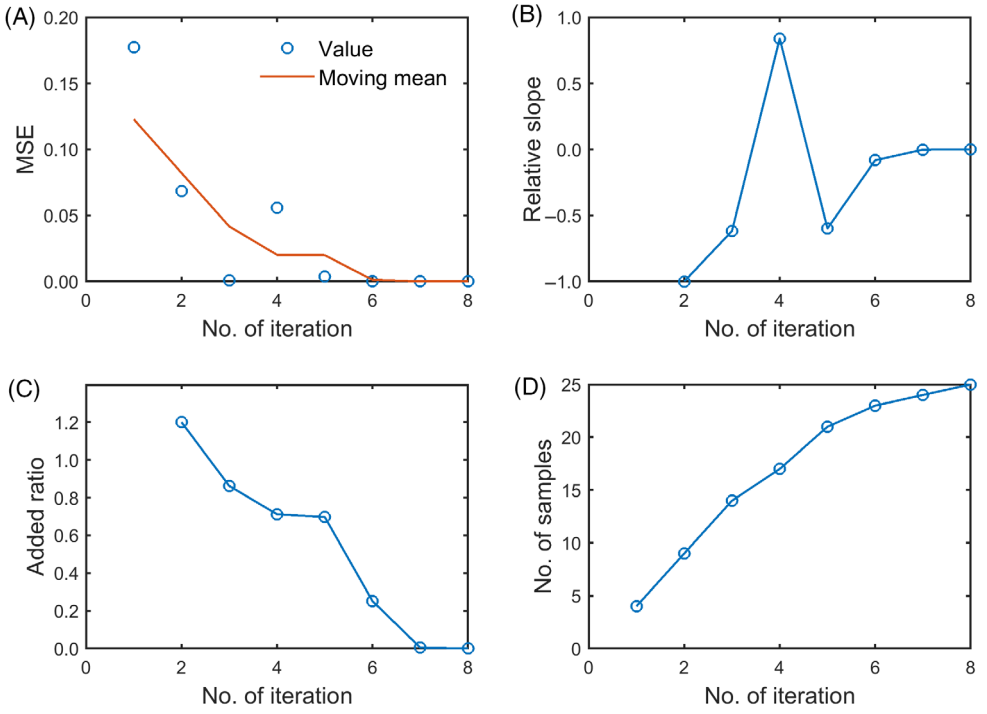

FIGURE 5 Slowdown sampling for the surrogate construction of series reaction kinetics

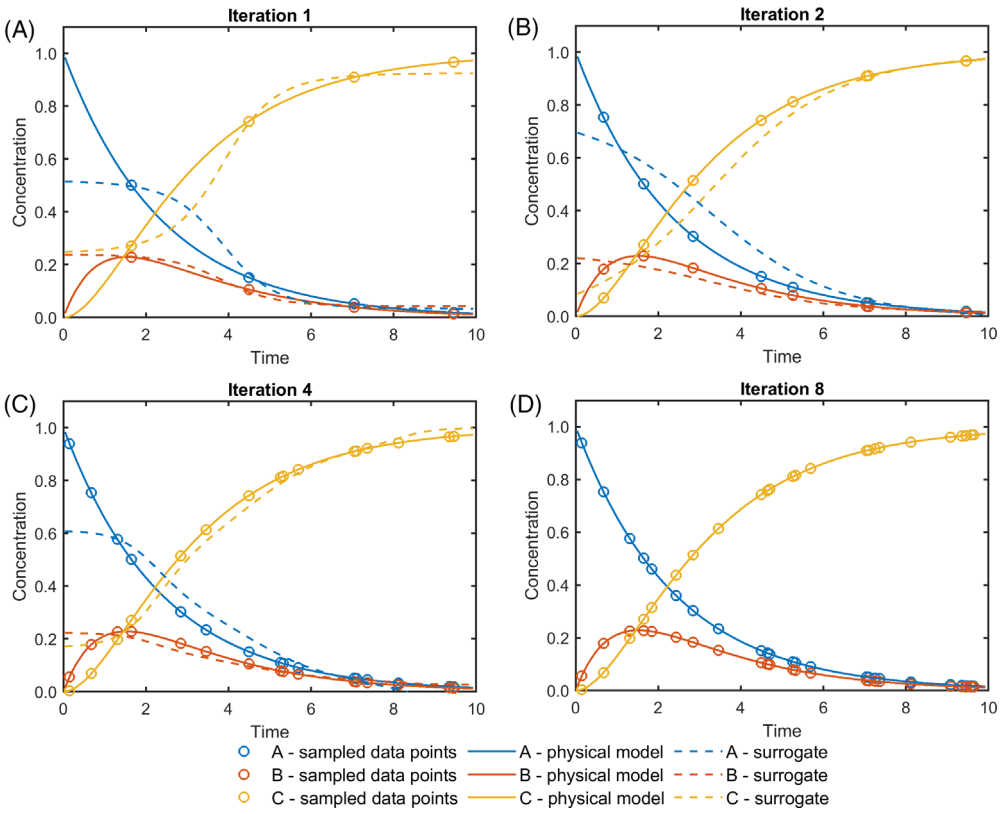

FIGURE 6 The regression performance of ANN surrogate for the concentration profiles of three species regarding the series reaction. Each iteration adds new data points to refine the surrogate. The performance of surrogate gradually improves from Iteration 1, 2, 4 to Iteration 8. Solid lines for the simulation by the physical model, while dashed lines for the simulation by the surrogate model. Surrogate is built based on the sampled data points

$$\frac{dC}{dt} = k_2 B \tag{7}$$

Based on this physical model, the concentration profiles of the three species are simulated. An ANN-based surrogate is iteratively refined by sequential sampling. As Figure 5A indicates, with more data added, the fitting performance improves (MSE decreases). Meanwhile, the decreasing rate of MSE becomes slower (Figure 5A) and the absolute value of relative slope tends to be smaller (Figure 5B). Following this, the added ratio decreases (Figure 5C) as well as the same trend is indicated for the sampling rate (Figure 5D). Once $\left|\text{slope}_{\text{relative}_i}\right| < 0.02$, the algorithm is terminated and collects 25 data points in total.

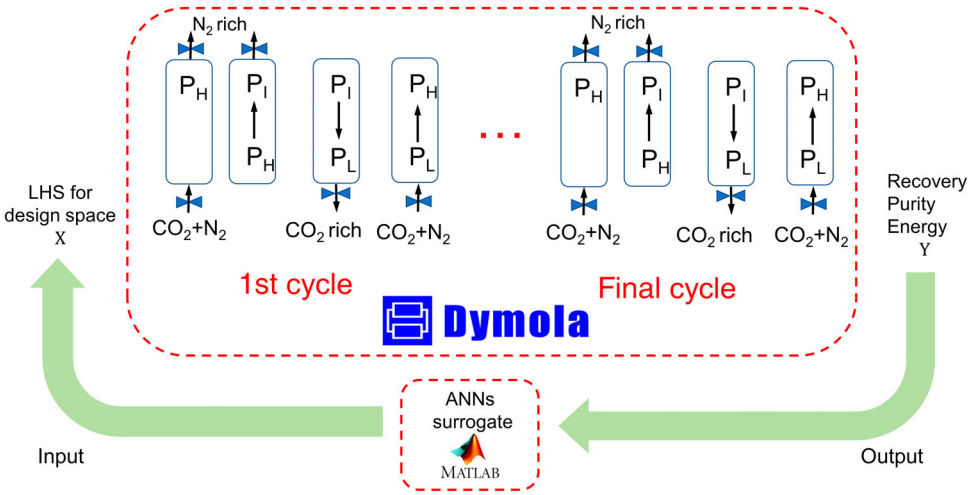

FIGURE 7 Surrogate construction of the four-stage PSA for CO_2 capture

TABLE 1 Description of input and output variables for the PSA surrogate

	Range	Unit	Notes
Input variables			
t_{ads}	20–100	[s]	Duration of adsorption stage
t_{bd}	30–200	[s]	Duration of blowdown stage
t_{evac}	30–200	[s]	Duration of evacuation stage
P_I	0.07–0.5	[100 kPa]	Setpoint of intermediate pressure
P_L	0.005–0.05	[100 kPa]	Setpoint of low pressure
v_{feed}	0.1–2	[m/s]	Inlet flowrate
y_{CO_2}	0.02–0.06	[–]	Inlet molar fraction of CO_2
Output variables			
Recovery		[–]	Recovery rate of CO_2
Purity		[–]	Purity of CO_2 in the product flow
Energy		[kWh/ton-CO_2]	Energy usage per ton CO_2 captured

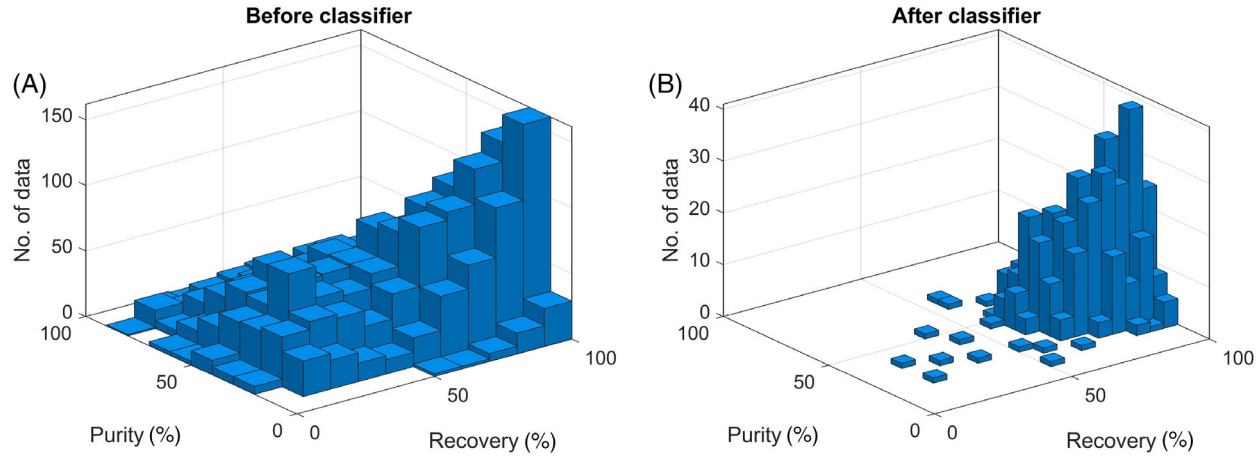

FIGURE 8 (A,B) Classification performance for PSA

To further evaluate the performance of surrogate, we simulate the concentration profiles of three species using physical model and surrogate model, respectively. Figure 6 shows that the regression performance of the iteratively refined surrogate gradually improves with iteration. The surrogate obtained in the final iteration (Iteration 8) can perfectly model the original concentration profiles of the three species.

3.2.3 | Discussion on slowdown sampling

The slowdown sampling maintains a good balance between training and sampling. In each iteration, a small number of networks are recommended to test. We consider two extremes. (a) When data is sufficient, slowdown sampling tends to give a small sampling rate. As a result, the number of total samples does not significantly change, while training is still performed in every iteration. This can be equivalent to an extreme situation, where sampling stops but excessive trainings are executed based on the sufficient data. (b) By contrast, when the data is insufficient in the initial iterations, slowdown sampling tends to deliver a large sampling rate, so fewer trainings but more samplings are executed in the initial iterations. Such a balance between training and sampling is automatically built by relating the improvement rate of surrogate ($slope_{MSE_i}$) to the sampling rate (N_{added_i}). However, this balance advantage is not obvious in this work because we focus on the case studies, where the computational cost on data generation is much more expensive than surrogate training.

The slowdown sampling can be well reproduced, which can be referred to an example of peaks function in Figures S2 and S3). Sequential sampling is performed four times on the peaks: the sampling trends are similar for the four times and the number of total sampled data points are close to each other (between 190 and 220).

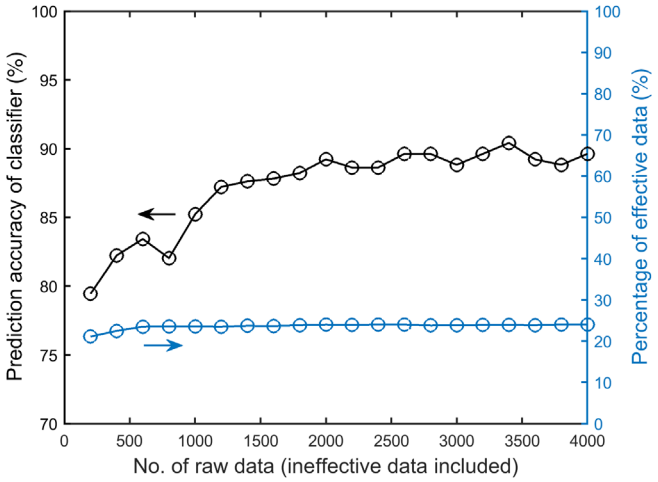

FIGURE 9 Effect of the number of raw data points on the classification performance for PSA. Raw data includes both effective data and ineffective data. Given the 24% effective data, 4000 raw data points result in 960 effective data points, which corresponds to the first iteration in the slowdown sampling

4 | CASE STUDIES

Two case studies come from two processes in carbon capture and utilization (CCU): PSA and gas-to-liquids (GTL), which starts from combined reforming (steam + CO_2) of natural gas.

4.1 | Case study 1: surrogate generation for PSA, a dynamic process built in Dymola

PSA is a cyclic dynamic process for gas separation. As to achieve the objective of net-zero 2050, PSA is regarded as a promising technology for CO_2 capture from fossil fuel-based processes.[43–45] Through continuously varying pressure, adsorption switches with desorption for all

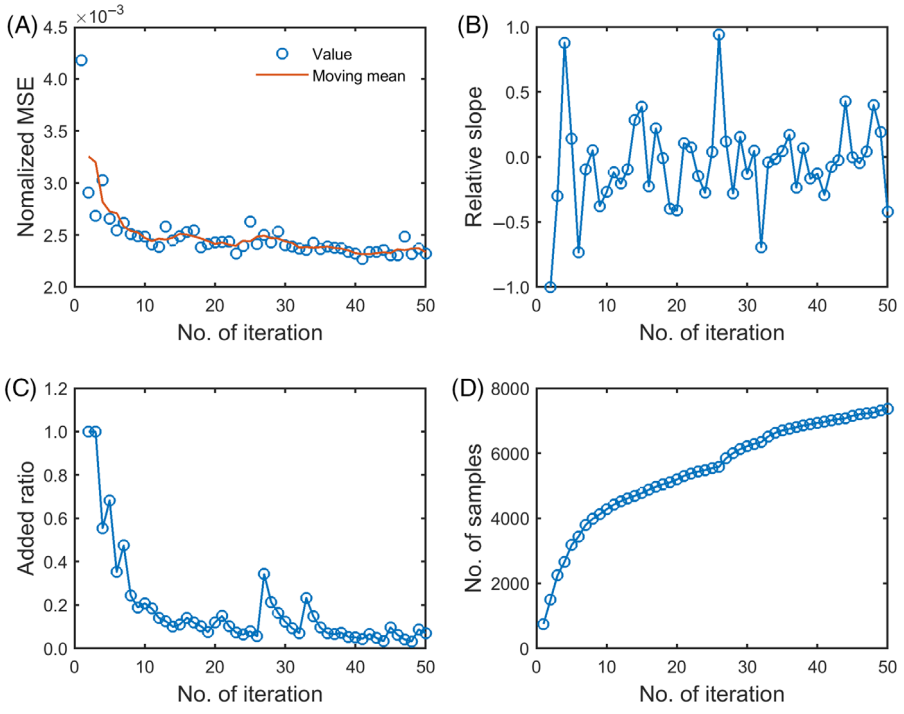

FIGURE 10 (A–D) Slowdown sampling for the surrogate construction of PSA

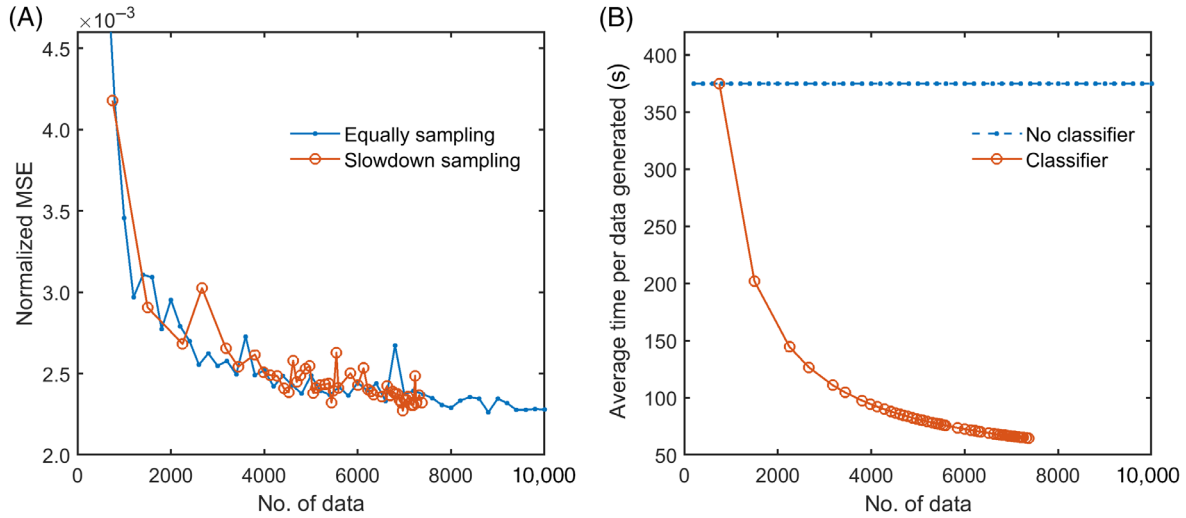

FIGURE 11 (A,B) The contribution of slowdown sampling and classifier for the efficiency improvement of PSA surrogate construction

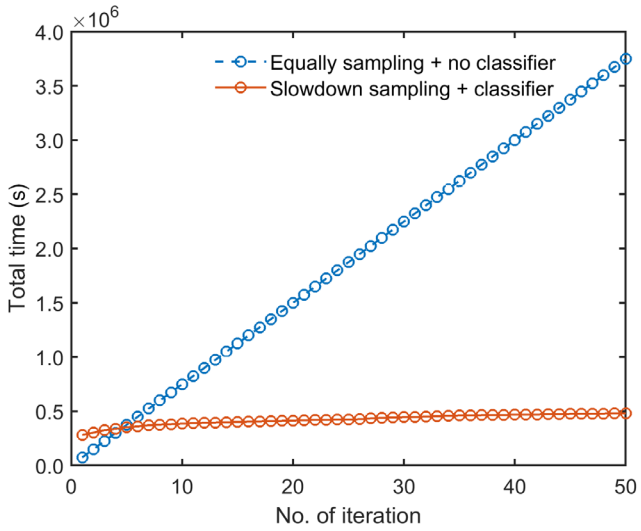

FIGURE 12 Comparison of total time spent on surrogate generation for PSA between [equally sampling + no classifier] and [slowdown sampling + classifier]. Total time is the sum of time spent on data generation and surrogate training

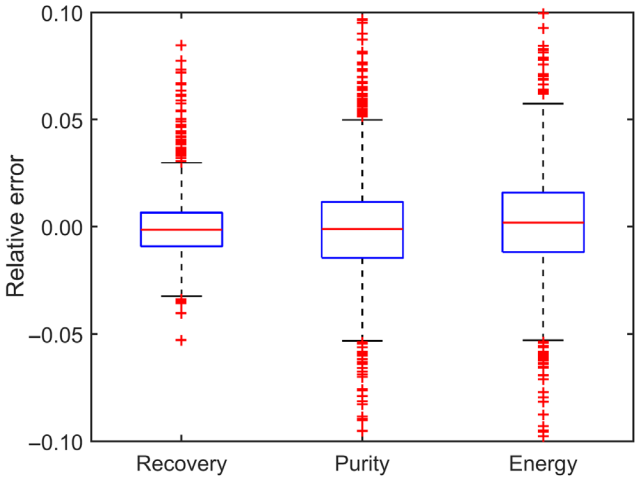

FIGURE 13 Prediction performance of the final surrogate for PSA

the process periods (Figure 7). Eventually, PSA reaches a cyclic steady state (CSS), where consecutive cycles have the same profile.

The physical model, dynamic simulation and a typical input–output data point of PSA are presented in the Section S3 of Supplementary Information. The physical model of PSA is complex due to its stiff and nonlinear partial differential equations (PDEs). The simulation based on the PSA physical model is time-demanding because the simulation result is only meaningful under CSS. In other words, one PSA simulation usually needs to be executed for a long time to reach CSS and then obtain one meaningful simulation output. A surrogate can solve the issues mentioned above, but we still need to minimize the computational cost of data generation to build the surrogate. We program the physical model of PSA in Dymola and use MATLAB to run Dymola to collect inputs/outputs dataset automatically.[38] Table 1 describes the inputs and outputs for the PSA system.

In this case, PSA is applied to capture the CO_2 from the flue gas of a natural gas power plant. Due to the low CO_2 concentration in the flue gas (~4%), one PSA unit cannot guarantee the required purity (~90% for carbon capture). PSA in series can be an option. In this work, we mainly focus on performance of the first PSA unit, where recovery of CO_2 is supposed to be high enough. The purity of CO_2 should improve as well. A trade-off relationship is reported between recovery and purity,[38,43] so the CO_2 purity cannot be too high given the priority on recovery. Therefore, we trained an SVM classifier to select the sample inputs, which are predicted to achieve a high recovery (higher part of distribution, >50%) and a moderate purity (middle part of distribution, 25%–75%). The classifier's performance can be referred to in Figure 8, and eventually, only 24% of the initial-sampled data is selected to fall in the desired space.

TABLE 2 Description of input and output variables for the GTL surrogate

	Range	Unit	Notes
Input variables			
F_{CO_2}	72–8200	[kmol/h]	Inlet flowrate of CO_2
F_{NG}	Design spec.[a]	[kmol/h]	Inlet flowrate of natural gas (NG)
x_{CH_4}	0.94–0.96	[–]	Inlet molar fraction of CH_4 (uncertainty)
T_{FT}	215–265	[°C]	Temperature in FT reactor
P_{FT}	15–50	[100 kPa]	Pressure in FT reactor
N_{trays} (integer)	45–65	[–]	No. of trays in distillation column
$T_{reformer}$	750–1000	[°C]	Temperature in reformer reactor
$P_{reformer}$	3–7	[100 kPa]	Pressure in reformer reactor
$Split_{vent}$	0.001–0.2	[–]	Split fraction to vent stream (the other to recycle)
$Split_{FT}$	0.01–0.99	[–]	Split fraction to FT (the other to reformer)
Output variables			
$F_{gasoline}$		[kmol/h]	Product flowrate of gasoline
F_{diesel}		[kmol/h]	Product flowrate of diesel
F_{gas}		[kmol/h]	Product flowrate of light HCs [C_1–C_4]
$F_{H_2O_{net}}$		[kmol/h]	Net flowrate of process water
$vent_{CO_2}$		[kmol/h]	Flowrate of CO_2 in the vent
Electricity		[MW]	Electricity usage for pumps and compressors
U_{air}		[GJ/h]	Cooling utility by air
U_{1000}		[GJ/h]	Heating utility by 1000°C fuel gas
U_{steam}		[GJ/h]	Heating utility by high pressure steam
U_{water}		[GJ/h]	Cooling utility by cooling water

[a]In Aspen Plus, F_{NG} is determined by a flowsheet option (design specification). Varying F_{NG} and steam flowrate can achieve the desired syngas ratio, where H_2:CO = 2–2.2 is preferred for the FT reaction.

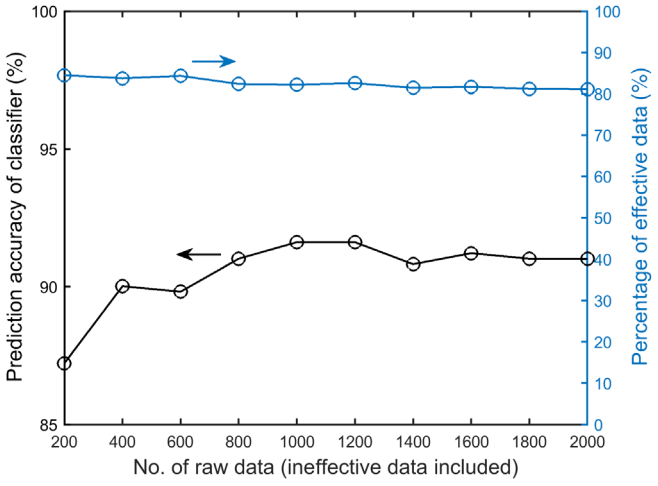

FIGURE 14 Effect of the number of raw data points on the classification performance for GTL. Thousand raw data (82% desired) and 2000 raw data (81% desired) correspond to the first two iterations, respectively, in slowdown sampling

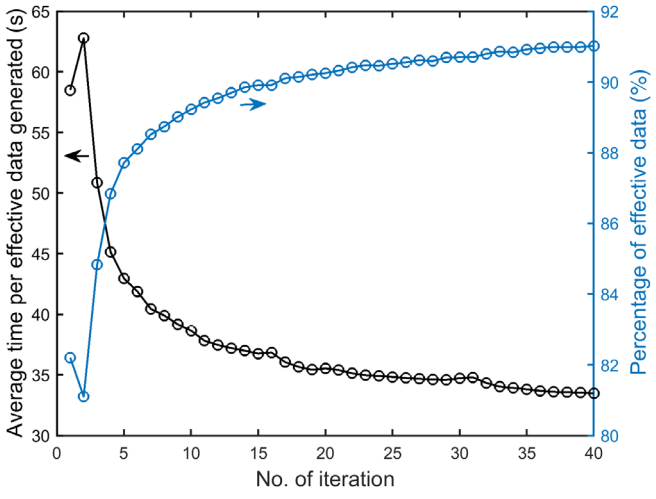

FIGURE 15 Improvement of data effectiveness by the classifier for GTL

Performance of SVM classifier looks imperfect, since some selected sample inputs still lead to undesired outputs. For examples, a few filtered samples have a recovery smaller than 50% (Figure 8B). Yet, this result is good-enough when referring to the prediction accuracy by the classifier (Figure 9). Raw data, containing effective data (desired) and ineffective data (undesired), is used to train SVM classifier. The training–test split is used to calculate the accuracy of prediction: 90% raw data is used to train the SVM classifier, while the other 10% raw data is used to examine its prediction performance. As

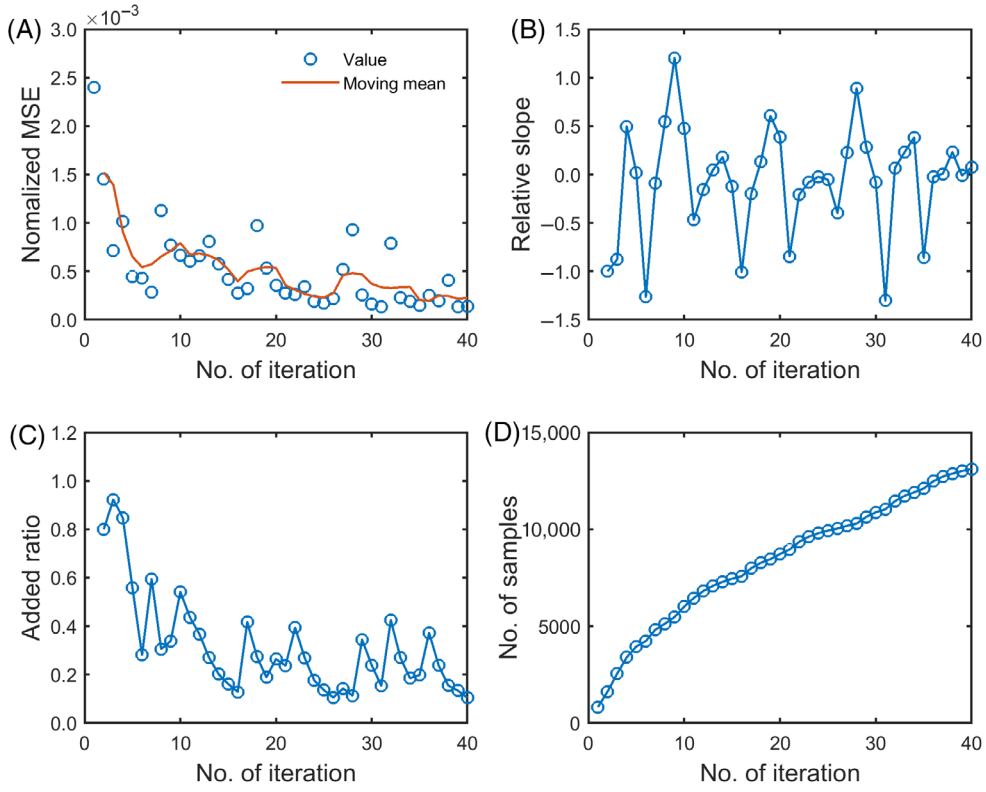

FIGURE 16 (A–D) Slowdown sampling for the surrogate construction of GTL

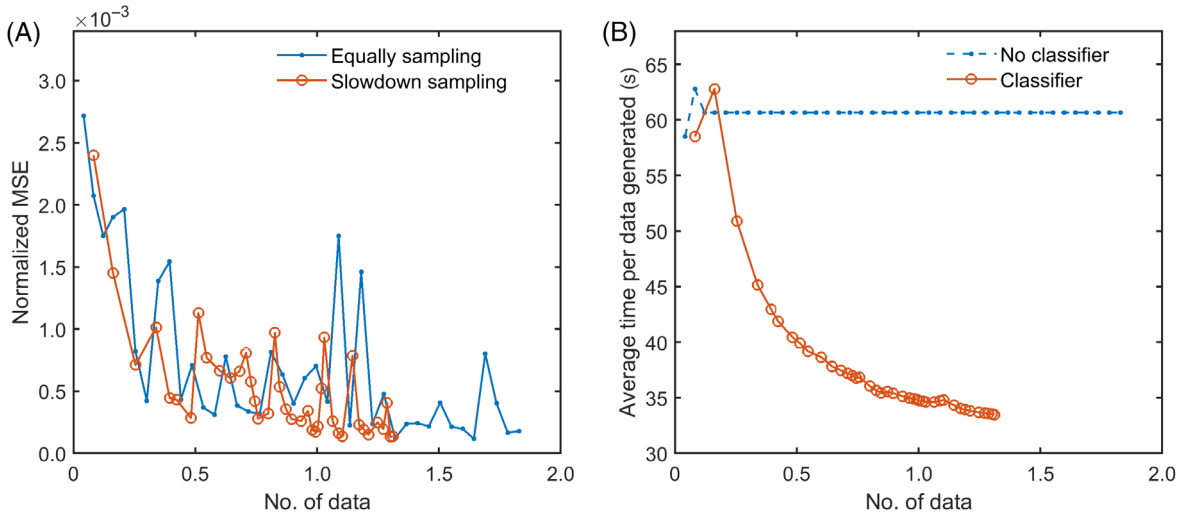

FIGURE 17 (A,B) The contribution of slowdown sampling and classifier for the efficiency improvement of GTL surrogate construction: slowdown sampling has much a higher possibility of collecting fewer data points than equal sampling; a classifier can reduce the average time per data generated. Clarification for the dashed line in b: since no classifier is used in the first two iterations, we assume that their average value for a single data generation will be the time in the successive iterations

shown in Figure 9, increasing the number of raw data can improve prediction accuracy to 90%. With more than 2000 data points, we see negligible improvement until 4000 data points. Since 90% accuracy is already good enough for a classifier, we stop sampling ineffective inputs/outputs for the purpose of SVM training. That is to say, after 4000 raw data points, SVM classifier commences its filtering function for the newly sampled inputs. To clarify the relationship between SVM classifier with the slowdown sampling, 4000 raw data points (24% desired) only contains 960 effective data points, which initializes the first iteration of the slowdown sampling.

The slowdown sampling is applied to collect effective data iteratively. Figure 10 indicates that the regression improvement is not significant after 10 iterations (Figure 10A), and the corresponding sampling rate gradually decreases in the meanwhile (Figure 10D). The

relative slope fluctuates significantly along with the iteration (Figure 10B), while the added ratio function helps smooth the fluctuation (Figure 10C). Eventually, we terminate the algorithm after 50 iterations, since the MSE value hardly decreases after the 40th iteration.

Efficiency of the proposed workflow can be demonstrated by comparing with a reference method with no classifier and with a static sampling rate (i.e., equally sampling at 200 data points per iteration). Although the initial rate of slowdown sampling is over three times that of equal sampling, the sampling rate keeps dropping and, eventually, falls below 50 data/iteration after the 40th iteration. Within 50 iterations, slowdown sampling generates 7372 samples, while equal sampling generates 10,000 samples. Notably, Figure 11A indicates that slowdown sampling has a much higher possibility for earlier termination. When a similar fitting performance is reached (e.g., MSE = 2.2E − 3), much fewer data points are collected by the slowdown sampling (6967 data points) than by equally sampling (8800 data points). Figure 11B illustrates the effect of the SVM classifier. The classier is trained by the dataset in the first iteration. The average time per data generated is assumed to be kept the same as the 1st iteration if no classifier applies (as the dashed line in Figure 11B). Herein, the data effectiveness without a classifier is around 24%. By contrast, the classifier can significantly improve the data quality by avoiding undesired inputs for the data generation, thus reducing the time per data generation by 83%, from 375 s (1st iteration) to 65 s (50th iteration).

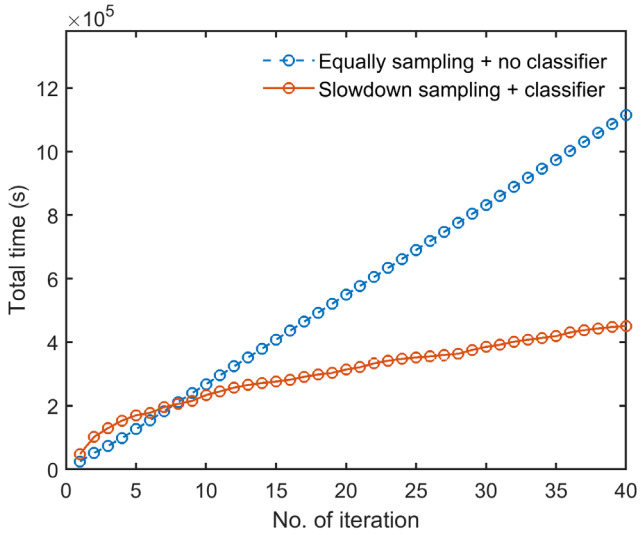

FIGURE 18 Comparison of total time spent on surrogate generation for GTL between [equally sampling + no classifier] and [slowdown sampling + classifier]. Total time is the sum of time spent on data generation and surrogate training

The effect of the two principles can be merged to improve efficiency of surrogate generation for PSA. Since fluctuations exists through sequential iterations, a termination is hard to be determined. Herein, we terminate the algorithms after 50 iterations. As shown in Figure 12, if equally sampling without classifier is applied, the time spent on surrogate generation for PSA is 3.8E + 6 s (50th iteration), which can be reduced by 87% if the two principles apply (4.8E + 5 s, 50th iteration). It might be unfair to compare based on the number of iterations, because a deviation can occur when the sampling rate for equal sampling changes. A reasonable comparison criterion can be based on a key iteration, which identifies the surrogate with the best regression performance. Based on the found minimal MSE = 2.2E − 3, the [slowdown sampling + classifier] requires 4.7E + 5 s (6967 data points, 41st iteration), while [equally sampling + no classifier] requires 3.3E + 6 s (8800 data points, 44th iteration). Hence, the proposed workflow can reduce the total time by 86%.

A separate dataset is used to test the performance of the iteratively refined surrogate for PSA. We employ the boxplot for the relative errors between the surrogate predictions and the rigorous simulations for the three outputs—recovery of CO_2, purity of CO_2 in

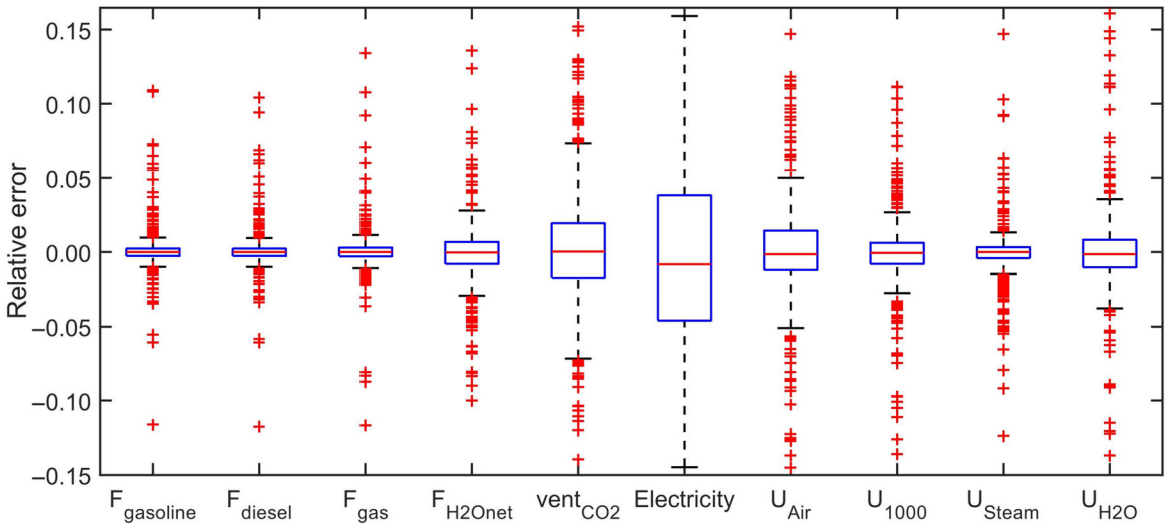

FIGURE 19 Prediction performance of the final surrogate for GTL

the product flow, and energy consumption of the system. As shown in Figure 13, most outputs can be predicted with relative errors smaller than 5%.

4.2 | Case study 2: Surrogate generation for GTL, a steady-state flowsheet in Aspen Plus

GTL is a classical chemical process for production of fuels.[46,47] We built a flowsheet in Aspen Plus (detailed information in Figure S6). The process starts with the combined reforming (steam + CO_2) of natural gas to syngas, followed by Fischer–Tropsch (FT) synthesis for fuels. A recycle stream is split: one for reforming, the other for FT rector.

The combined reforming, FT (kinetics and chain growth probability for products distribution), simulation (convergence) and a typical input–output data point of GTL system are presented in the Section-S4 of Supporting Information. To seek the optimal operating condition, we may optimize some decision variables under uncertainty (input for surrogate) to evaluate the corresponding process performance (output for surrogate) as shown in Table 2.

Aspen Plus simulations suffer from nonconvergence issues when improper operating conditions are given, or the recycle stream is set too tight.[30,48] Such problems also occur in our case. SVM classifier is employed to avoid the nonconvergence issues. As shown in Figure 14, 200 raw data can deliver a good classifier with an accuracy at 87%, which can further increase to 91% with 2000 raw data points. After the 2000 raw data points, the SVM classifier commences its filtering function for the newly sampled inputs. To clarify the relationship between SVM classifier with the slowdown sampling, 1000 raw data (82% desired) and 2000 raw data (81% desired) correspond to the first two iterations, respectively, in slowdown sampling.

The obtained classifier is applied here to screen out the potential nonconverged inputs in the successive iterations, thus improving the percentage of effective data from 81% to 91% (Figure 15). Notably, the nonconverged simulations in Aspen Plus usually takes a long time to stop but deliver invalid outputs. A 10% improvement for effective data tremendously cut the time per data generation by 46%, from 61 s (1–2 iterations) to 33 s (40th iteration).

The slowdown sampling is applied to collect data iteratively. The relative slope in Figure 16B fluctuates more significantly than the case study of PSA, see Figure 10. That is probably because the GTL has more outputs to fit, and the regression is more complex than the case of PSA. The observed trend in Figure 16A indicates that the regression improvement is not significant after the 25th iteration. Eventually, we terminate the workflow after 40 iterations to avoid the unnecessary computational costs.

The efficiency of the proposed workflow can be demonstrated by comparison to a reference method with no classifier and equal sampling (a slow static sampling). The two principles can separately improve the sampling efficiency for building surrogates for GTL. As shown in Figure 17A, the slowdown sampling has a higher chance for an earlier termination than the equally sampling, to achieve a similar fitting performance (MSE = 1E − 4) with fewer data points (slowdown for 11,000 data points vs. equally sampling for 13,200 data points). The trend in Figure 17B shows that the SVM classifier can reduce the average time spent on individual points by 46%.

Overall, the effect of two principles can be merged to improve the efficiency of surrogate generation for GTL. As shown in Figure 18, based on the found minimal MSE = 1E − 4, the [slowdown sampling + classifier] requires 3.9E + 6s (11,000 data points, 31st iteration), while [equally sampling + no classifier] requires 8.0E + 6s (13,200 data points, 29th iteration). Hence, the proposed workflow can reduce the total time by 51%.

A separate test dataset is used to evaluate the performance of the surrogate obtained in the final iteration. We employ the boxplot for the relative errors between the surrogate predictions and the rigorous simulations for the 10 outputs. As shown in Figure 19, most outputs can be well predicted with relative errors smaller than 5%, and some are even smaller than 1%, for example, the mass flowrate for the fuel products. The fitting for the utility is not ideal, and the relative error of the electricity consumption can go up to 15%. This is probably due to the insufficient feature selection for utility fitting. For example, the electricity consumption is related to the units of pumps and compressors, while no relevant features are selected into the inputs for the surrogate training. Meanwhile, no features related to heat exchangers are chosen, so the fitting performance of the utility is not as good as the mass flowrates. However, the motivation behind surrogate is to build a reduced-order model as to replace the original full-order physical model, and thus sacrificing partial accuracy is unavoidable but acceptable.

5 | CONCLUSIONS AND OUTLOOK

This work develops an efficient workflow for the surrogate generation for engineering systems (typically $t_{data} \gg t_{training}$). The efficiency benefits from the improvement in data quality and the reduction in data quantity. (a) A classifier is trained to avoid the undesired design space for data generation and improve the data quality. To train a good-enough classifier (over 90% accuracy) requires a relatively small amount of dataset, which can work as the data source for the initial iteration of slowdown sampling. The obtained SVM classifiers can dramatically cut the computational cost per data generation by 83% for PSA and 46% for GTL. (b) A slowdown sampling employs a dynamic sampling rate: initially sampling is fast to collect nearly sufficient amount of data in just a few iterations, and gradually slows down with the improvement of surrogate. The slowdown sampling can spot the nonimprovement trend for the surrogate quality at a relatively early stage, which thus lowers the possibility for oversampling (data quantity). With the proposed workflow, the computational costs of surrogate generation is shown to be reduced by 86% for PSA and 51% for GTL case studies, compared with that by employing a static sampling rate to achieve a similar standard of surrogate. Technically, our

methodology is straightforward to implement because no intensive mathematical formulations are involved.

Notably, the proposed workflow can be generalized to other surrogate types and it should be compatible to the other existing sampling methods. The exploitation-based methods can be introduced to integrate with our workflow, as to properly increase sampling probability in the nonlinear/complex design space. The primary goal of this work was to investigate the influence of the sampling rate for the surrogate generation. Thus, the sampling was desired to be a homogenous type, which might be disturbed by exploitation-based methods. As a result, we only considered exploration-based methods in our current workflow. Another work that can be done is to determine proper termination criteria: we tried to stop the algorithm when the MSE difference between two consecutive iterations approached 0, or the slope approached 0, but the fluctuation of MSE values always existed for the case study of GTL or PSA, which made the tolerance value for termination hard to set. One possible solution is to apply feature selection techniques (i.e., automatically adjust input variables) to improve fitting performance and reduce the fluctuation during sequential sampling.

Additionally, this work lays foundation for the digitalization and superstructure optimization of an extensive CCU system. The two case studies presented in this work belong to its two process options. The subsystems of CCU are usually modeled in different process simulators, which cause inconvenience for overall simulation or optimization. This work enables the representation of CCU with the machine learning-based digital twins, following by overall optimization in a high-level interactive platform.

ACKNOWLEDGMENTS

ZH and CZ acknowledge financial support from Chinese Scholarship Council and Cambridge Trust. ZH's final-year PhD was partially funded by the Sustainable Reaction Engineering research group of Prof. Lapkin. AAL acknowledges funding from the National Research Foundation (NRF), Prime Minister's Office, Singapore, under its Campus for Research Excellence and Technological Enterprise (CREATE) program as a part of the Cambridge Centre for Advanced Research and Education in Singapore Ltd (CARES Ltd).

CONFLICT OF INTEREST

Authors declare no competing financial interest.

AUTHOR CONTRIBUTIONS

Zhimian Hao: Conceptualization (lead); investigation (lead); methodology (lead); software (lead); writing – original draft (lead); writing – review and editing (lead). **Chonghuan Zhang:** Methodology (supporting); software (supporting); writing – review and editing (supporting). **Alexei Lapkin:** Conceptualization (lead); funding acquisition (lead); supervision (lead); writing – review and editing (supporting).

DATA AVAILABILITY STATEMENT

Data available on request from the authors

ORCID

Alexei A. Lapkin https://orcid.org/0000-0001-7621-0889

REFERENCES

1. Lasi H, Fettke P, Kemper H-G, Feld T, Hoffmann M. Industry 4.0. *Bus Inf Syst Eng*. 2014;6(4):239-242.
2. Kenett RS, Bortman J. The digital twin in industry 4.0: a wide-angle perspective. *Qual Reliab Engng Int*. 2021;1-10. https://doi.org/10.1002/qre.2948
3. Garud SS, Karimi IA, Kraft M. Design of computer experiments: a review. *Comput Chem Eng*. 2017;106:71-95.
4. Kim SH, Boukouvala F. Machine learning-based surrogate modeling for data-driven optimization: a comparison of subset selection for regression techniques. *Optim Lett*. 2020;14(4):989-1010.
5. Cao L, Russo D, Felton K, et al. Optimization of formulations using robotic experiments driven by machine learning DoE. *Cell Rep Phys Sci*. 2021;2(1):100295.
6. Saripella KK, Loka NC, Mallipeddi R, Rane AM, Neau SH. A quality by experimental design approach to assess the effect of formulation and process variables on the extrusion and Spheronization of drug-loaded pellets containing Polyplasdone® XL-10. *AAPS PharmSciTech*. 2016;17(2):368-379.
7. Zhang C, Amar Y, Cao L, Lapkin AA. Solvent selection for Mitsunobu reaction driven by an active learning surrogate model. *Org Process Res Dev*. 2020;24(12):2864-2873.
8. Henao CA, Maravelias CT. Surrogate-based superstructure optimization framework. *AIChE J*. 2011;57(5):1216-1232.
9. Bhosekar A, Ierapetritou M. Advances in surrogate based modeling, feasibility analysis, and optimization: a review. *Comput Chem Eng*. 2018;108:250-267.
10. McBride K, Sundmacher K. Overview of surrogate modeling in chemical process engineering. *Chem Ing Tech*. 2019;91(3):228-239.
11. del Rio-Chanona EA, Fiorelli F, Zhang D, Ahmed NR, Jing K, Shah N. An efficient model construction strategy to simulate microalgal lutein photo-production dynamic process. *Biotechnol Bioeng*. 2017;114(11):2518-2527.
12. Boukouvala F, Ierapetritou MG. Surrogate-based optimization of expensive flowsheet modeling for continuous pharmaceutical manufacturing. *J Pharm Innov*. 2013;8(2):131-145.
13. Wan X, Pekny JF, Reklaitis GV. Simulation based optimization of supply chains with a surrogate model. In: Barbosa-Póvoa A, Matos H, eds. *Computer Aided Chemical Engineering*. Vol 18. Elsevier; 2004:1009-1014.
14. Ye W, You F. A computationally efficient simulation-based optimization method with region-wise surrogate modeling for stochastic inventory management of supply chains with general network structures. *Comput Chem Eng*. 2016;87:164-179.
15. Yondo R, Andrés E, Valero E. A review on design of experiments and surrogate models in aircraft real-time and many-query aerodynamic analyses. *Prog Aerosp Sci*. 2018;96:23-61.
16. Simpson TW, Mauery TM, Korte JJ, Mistree F. Kriging models for global approximation in simulation-based multidisciplinary design optimization. *AIAA J*. 2001;39(12):2233-2241.
17. Queipo NV, Haftka RT, Shyy W, Goel T, Vaidyanathan R, Kevin TP. Surrogate-based analysis and optimization. *Prog Aerosp Sci*. 2005;41(1):1-28.
18. Forrester AIJ, Keane AJ. Recent advances in surrogate-based optimization. *Prog Aerosp Sci*. 2009;45(1):50-79.
19. Fahmi I, Cremaschi S. Process synthesis of biodiesel production plant using artificial neural networks as the surrogate models. *Comput Chem Eng*. 2012;46:105-123.
20. Duvenaud D. *Automatic Model Construction with Gaussian Processes*. University of Cambridge; 2014.
21. Bradford E, Schweidtmann AM, Lapkin A. Efficient multiobjective optimization employing Gaussian processes, spectral sampling and a genetic algorithm. *J Global Optim*. 2018;71(2):407-438.

22. Schweidtmann AM, Mitsos A. Deterministic global optimization with artificial neural networks embedded. *J Optim Theory Appl*. 2019; 180(3):925-948.
23. Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators. *Neural Netw*. 1989;2(5):359-366.
24. Boyle P, Frean M. *Multiple Output Gaussian Process Regression*. Victoria University of Wellington; 2005.
25. Wilson AG, Knowles DA, Ghahramani Z. Gaussian process regression networks. *arXiv*. 2011. https://arxiv.org/abs/1110.4411
26. Sircar S. Pressure swing adsorption. *Ind Eng Chem Res*. 2002;41(6): 1389-1392.
27. Jasra RV, Choudary NV, Bhat SGT. Separation of gases by pressure swing adsorption. *Sep Sci Technol*. 1991;26(7):885-930.
28. Jee J-G, Lee J-S, Lee C-H. Air separation by a small-scale two-bed medical O_2 pressure swing adsorption. *Ind Eng Chem Res*. 2001; 40(16):3647-3658.
29. Eiben AE, Smith JE. *Introduction to Evolutionary Computing*. Vol 53. Springer; 2003.
30. Ibrahim D, Jobson M, Li J, Guillén-Gosálbez G. Surrogate models combined with a support vector machine for the optimized design of a crude oil distillation unit using genetic algorithms. In: Espuña A, Graells M, Puigjaner L, eds. *Computer Aided Chemical Engineering*. Vol 40. Elsevier; 2017:481-486.
31. Kim Y, Kim E, Antono E, Meredig B, Ling J. Machine-learned metrics for predicting the likelihood of success in materials discovery. *npj Comput Mater*. 2020;6(1):131.
32. Houben C, Peremezhney N, Zubov A, Kosek J, Lapkin AA. Closed-loop multitarget optimization for discovery of new emulsion polymerization recipes. *Org Process Res Dev*. 2015;19(8):1049-1053.
33. Cozad A, Sahinidis NV, Miller DC. Learning surrogate models for simulation-based optimization. *AIChE J*. 2014;60(6):2211-2227.
34. Wilson ZT, Sahinidis NV. The ALAMO approach to machine learning. *Comput Chem Eng*. 2017;106:785-795.
35. Busby D, Farmer CL, Iske A. Hierarchical nonlinear approximation for experimental design and statistical data fitting. *SIAM J Sci Comput*. 2007;29(1):49-69.
36. Li G, Aute V, Azarm S. An accumulative error based adaptive design of experiments for offline metamodeling. *Struct Multidiscipl Optim*. 2010;40(1-6):137-155.
37. Lee U, Burre J, Caspari A, Kleinekorte J, Schweidtmann AM, Mitsos A. Techno-economic optimization of a green-field post-combustion CO_2 capture process using superstructure and rate-based models. *Ind Eng Chem Res*. 2016;55(46):12014-12026.
38. Hao Z, Caspari A, Schweidtmann AM, Vaupel Y, Lapkin AA, Mhamdi A. Efficient hybrid multiobjective optimization of pressure swing adsorption. *Chem Eng J*. 2021;423:130248.
39. Sahraei MH, Ricardez-Sandoval LA. An integration framework for CO_2 capture processes. In: Papadopoulos AI, Seferlis P, eds. *Process Systems and Materials for CO_2 Capture*. Wiley; 2017: 523-543.
40. Mahapatra P, Ma J, Ng B, Bhattacharyya D, Zitney SE, Miller DC. Integrated dynamic modeling and advanced process control of carbon capture systems. *Energy Procedia*. 2014;63:1354-1367.
41. Cortes C, Vapnik V. Support-vector networks. *Mach Learn*. 1995; 20(3):273-297.
42. Nalepa J, Kawulok M. Selecting training sets for support vector machines: a review. *Artif Intell Rev*. 2019;52(2):857-900.
43. Haghpanah R, Majumder A, Nilam R, et al. Multiobjective optimization of a four-step adsorption process for postcombustion CO_2 capture via finite volume simulation. *Ind Eng Chem Res*. 2013;52(11): 4249-4265.
44. Leperi KT, Yancy-Caballero D, Snurr RQ, You F. 110th anniversary: surrogate models based on artificial neural networks to simulate and optimize pressure swing adsorption cycles for CO_2 capture. *Ind Eng Chem Res*. 2019;58(39):18241-18252.
45. Liu LY, Gong H, Wang Z, Li G, Du T. Application of pressure swing adsorption technology to capture CO_2 in highly humid flue gas. *Prog Chem*. 2018;30(6):872-878.
46. Dry ME. The Fischer–Tropsch process: 1950–2000. *Catal Today*. 2002;71(3):227-241.
47. van Ommen JR, Grievink J. Synthesis gas utilization for transportation fuel production. In: van Ommen JR, de Jong W, eds. *Biomass as a Sustainable Energy Source for the Future*. Wiley; 2014: 525-546.
48. Penteado AT, Esche E, Weigert J, Repke J-U. A framework for stochastic and surrogate-assisted optimization using sequential modular process simulators. In: Pierucci S, Manenti F, Bozzano GL, Manca D, eds. *Computer Aided Chemical Engineering*. Vol 48. Elsevier; 2020: 1903-1908.

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of the article at the publisher's website.

How to cite this article: Hao Z, Zhang C, Lapkin AA. Efficient surrogates construction of chemical processes: Case studies on pressure swing adsorption and gas-to-liquids. *AIChE J*. 2022;e17616. doi:10.1002/aic.17616