DATA-DRIVEN MIXED-INTEGER OPTIMIZATION FOR MODULAR PROCESS INTENSIFICATION

A Dissertation
Presented to
The Academic Faculty

by

Sun Hye Kim

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Chemical and Biomolecular Engineering

Georgia Institute of Technology
December 2020

DATA-DRIVEN MIXED-INTEGER OPTIMIZATION FOR MODULAR PROCESS INTENSIFICATION

Approved by:

Dr. Fani Boukouvala, Advisor
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Dr. Andrew Medford
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Dr. Ryan Lively
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Dr. Matthew Realff
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Dr. Santanu Dey
School of Industrial and Systems Engineering
Georgia Institute of Technology

Date Approved: November 30th, 2020

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. Fani Boukouvala, for her guidance and support, which made it possible for this thesis to come along as one piece. I would like to also thank the rest of my thesis committee members for their insightful comments that allowed me to broaden my research.

In addition, I thank my friends for moral supports and for making my Ph.D. journey more enjoyable. While I will not miss exams and deadlines, I will definitely miss our mid-day coffee chats and all the fun we had in the last 4.5 years.

I deeply thank my parents and sister for their encouragement and supporting me throughout writing this thesis and my life in general. I would not be where I am today without their support. Last but not least, to my fiancé Boo: thanks for providing me encouragement and motivation to finish this journey (and also for feeding me lots of really good food). Without you believing in me, I never would have made it, so cheers to a new chapter!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

High-fidelity computer simulations provide accurate information on complex physical systems. These often involve proprietary codes, if-then operators, or numerical integrators to describe phenomena that cannot be explicitly captured by physics-based algebraic equations[1]. Consequently, the derivatives of the model are either absent or too complicated to compute; thus, the system cannot be directly optimized using derivative-based optimization solvers. Such problems are known as "black-box" systems since the constraints and the objective of the problem cannot be obtained as closed-form equations[1-7]. One promising approach to optimize black-box systems is surrogate-based optimization. Surrogate-based optimization uses simulation data to construct low-fidelity approximation models. These models are optimized to find an optimal solution.

We study several strategies for surrogate-based optimization for nonlinear and mixed-integer nonlinear black-box problems. First, we explore several types of surrogate models, ranging from simple subset selection for regression models to highly complex machine learning models. Second, we propose a novel surrogate-based optimization algorithm for black-box mixed-integer nonlinear programming problems. The algorithm systematically employs data-preprocessing techniques, surrogate model fitting, and optimization-based adaptive sampling to efficiently locate the optimal solution. Finally, a case study on modular carbon capture is presented. Simultaneous process optimization and adsorbent selection are performed to determine the optimal module design. An economic analysis is presented to determine the feasibility of a proposed modular facility.

CHAPTER 1. INTRODUCTION

Many optimization problems today depend on highly complicated computer simulations, which provide accurate and useful data that represent complex physical phenomena[5,8-12]. These simulations typically consist of a large system of equations, such as partial differential and ordinary differential equations, to accurately approximate physical systems. Due to the large size and complexity of the simulation or the presence of discontinuities caused by periodic boundary conditions, simulation-based optimization, also known as black/gray-box or derivative-free optimization, has recently gained significant attention[5,10,13-31] and can also be used for proprietary or legacy simulation codes.

Black-box (bb) optimization relies on data generated from complex simulations instead of first-principle models consisting of explicit analytical equations (Figure 1). Black-box problems have many applications in chemical engineering, such as parameter estimation for simulation-based systems[3,4,32], flowsheet synthesis[33], supply chain optimization[34,35], oilfield operations[36-39], and protein structure prediction[40-42]. Existing black-box optimization algorithms proposed in the literature can be divided broadly into three categories: sampling-based, surrogate-based, and stochastic or evolutionary methods[5,9,11]. Of the three aforementioned categories, the surrogate-based optimization literature has attracted significant attention lately, and this is undoubtedly linked to the recent developments in Machine Learning (ML)[43]. Many researchers from diverse fields have observed that surrogate-based optimization is a very promising method[1,44-46], and this is mainly due to the ability of the surrogate model to expedite the search for global optima while reducing the sampling requirements.

Figure 1. Illustration of a black-box problem; only the input and output data are available without the knowledge of the system

In this work, we address the following optimization problem (1):

$$\min_{x,y} f(\boldsymbol{x},\boldsymbol{y})$$

$$s.t.\, g_c(\boldsymbol{x},\boldsymbol{y}) \le 0, c = 1,\dots,C \tag{1}$$

$$x_i^l \le x_i \le x_i^u, x_i \in \mathbb{R}, y_j \in \{0,1\}, i = 1,..,I, j = 1,\dots,J$$

where x_i represents a continuous variable, x_i^l and x_i^u are lower and upper bounds of x_i, respectively; y_j represents a binary variable; $f(\boldsymbol{x},\boldsymbol{y})$ represents the objective function, and $g_c(\boldsymbol{x},\boldsymbol{y})$ represents a constraint with an unknown functional form, or a constraint that is embedded within the simulation and cannot be represented by a known algebraic equation. If only continuous variables exist (i.e., $J = 0$), (1) is a constrained nonlinear programming (NLP) problem; otherwise, (1) is a constrained mixed-integer nonlinear programming (MINLP) problem. We focus on solve (1) using surrogate-based optimization. Also known as a metamodel, a surrogate model uses simulation data to approximate the input-output relationship. Using surrogate modeling, we seek to obtain $\hat{f}(\boldsymbol{x},\boldsymbol{y})$ and $\hat{g}_c(\boldsymbol{x},\boldsymbol{y})$, which are low-fidelity approximations of $f(\boldsymbol{x},\boldsymbol{y})$ and $g_c(\boldsymbol{x},\boldsymbol{y})$.

Solving (1) using surrogate-based optimization is challenging due to various reasons. First, an optimal solution should be found with a limited number of function evaluations and computation time[7]. This is crucial especially when the simulation is time-consuming and/or the computation resource is limited. A strategic sampling and optimization approach is therefore required. Second, the physical system that needs to be approximated by a surrogate model tends to be highly complex, nonlinear, and nonconvex. Therefore, an efficient surrogate modeling strategy is needed to accurately approximate the physical system. Finally, the presence of discrete variables makes the search space discontinuous. Existing surrogate models assume the input space is ordinal, and the use of surrogate models for discrete search space has not been studied extensively.

In this work, we propose several algorithmic strategies to solve black-box NLP and MINLP problems and apply these strategies to design a modular carbon capture process. In chapter 2, we present some background information on black-box optimization with a specific focus on surrogate modeling and optimization. In chapter 3, we investigate the use of subset selection for regression techniques to create interpretable surrogate models for black-box NLP problems. An algorithmic framework to solve a black-box MINLP problem is presented in chapter 4. Chapter 5 presents a case study on modular carbon capture. An efficient strategy is proposed for simultaneous process optimization and adsorbent selection using a surrogate-based MINLP algorithm. Finally, we present a feasibility study on a modular facility by performing economic analysis. We end with conclusions and future perspective.

CHAPTER 2. BACKGROUND

2.1 Black-Box Optimization

Black-box optimization is also known as derivative-free optimization (DFO) or simulation-based optimization. It is used when the original derivative information is unavailable or difficult to obtain[11]. Unlike deterministic optimization, black-box optimization does not involve direct computation of derivatives for $f(\boldsymbol{x}, \boldsymbol{y})$ or $g_c(\boldsymbol{x}, \boldsymbol{y})$. Instead, different strategies are used to locate an optimal solution without the use of original derivative information. Existing black-box optimization algorithms can be divided into two broad categories: direct-search methods and model-based methods. First, literature reviews on these methods are briefly introduced for both NLPs and MINLPs. The rest of the chapter is dedicated to explaining surrogate-based optimization in detail.

2.1.1 Direct-Search Methods

The direct-search algorithm generates a set of neighborhood points and evaluates the values of the objective function at those points. The algorithm then either expands or narrows the search radius to explore other areas of the search space or to refine the solution. Some commonly used NLP direct-search algorithms are Nelder-Mead simplex algorithm[47], generalized pattern search[48], Mesh adaptive direct search[49], DIRECT algorithm[50], branch-and-bound, and multilevel coordinate search[51].

Direct-search MINLP algorithms are an extension to previously discussed NLP direct-search methods. Audet et al.[52] made modifications to the neighborhood point selection criterion to handle discrete variables, and the convergence to local optimality was

proved for both continuous and discrete variables. This work was further extended by Abramson et al.[53] to solve constrained mixed-variable optimization problems. Furthermore, Liuzzi et al.[54] modified a direct-search algorithm to solve MINLPs with both nonlinear and box constraints. The proposed algorithm alternates between local minimization of continuous variables and local search of discrete variables, and adaptive direct search is used to force convergence. The most widely used direct-search MINLP software is NOMAD (Nonsmooth Optimization by Mesh Adaptive Direct Search). However, NOMAD is only suitable for problems with cheap function evaluation as pattern-search algorithms tend to require a large number of samples to converge. Also, no extensive numerical studies currently exist for solving constrained mixed-integer problems with NOMAD[55].

2.1.2 Model-Based Methods

A model-based algorithm involves constructing an approximation model that represents the input-output relationship of the data. These approximation models are usually known as surrogate, meta-, or reduced-order models. Surrogate-based optimization has been used extensively for black-box NLPs, and it is generally divided into two categories: local and global models[11]. Local model-based search algorithms include trust-region method[56] and implicit filtering[57], where the surrogate model is developed for only a subset of the search space. For global model-based search algorithms, the surrogate model is constructed for the entire search region to guarantee convergence to global minimum[58].

Despite its success in solving NLP problems, surrogate-based optimization has only recently been applied to MINLP problems. Existing works on MINLPs usually rely on

solving an auxiliary mixed-integer problem[59,60], are not fully black-box (i.e. only a part of the problem formulation is black-box)[61], or can only handle binary values or low-dimensional problems. Some efforts have recently been made to overcome these limitations. SO-MI introduced in [55] uses a cubic RBF model and stochastic sampling strategies to solve expensive black-box problems. At each iteration, four groups of integer-feasible points are generated, and the best point is added to update the surrogate model. Similarly, MI-SO[62] alternates between local and global optimization search steps to locate high accuracy solutions. Although both algorithms have been tested on high-dimensional problems with non-binary discrete variables and do not require mixed-integer sub-solvers, it is heavily dependent on function evaluation and sampling. Also, the performance has only been evaluated with respect to the number of function evaluations; no remark on computation time is given in the paper.

2.2 Overview of Surrogate-based Optimization

Existing surrogate-based optimization algorithms generally have four steps: 1) initial sampling, 2) construction of a surrogate, 3) optimization, and 4) solution refinement (Figure 2). First, the construction of a surrogate model begins with choosing an efficient sampling strategy to maximize the information gained while minimizing the number of samples. Samples should be uniformly, but not regularly, distributed in the search space; i.e., if we project the sample points onto each variable axis, no projections of the sample points will overlap. Latin Hypercube sampling (LHS) is a widely used strategy to construct a space-filling, controlled random samples[7].

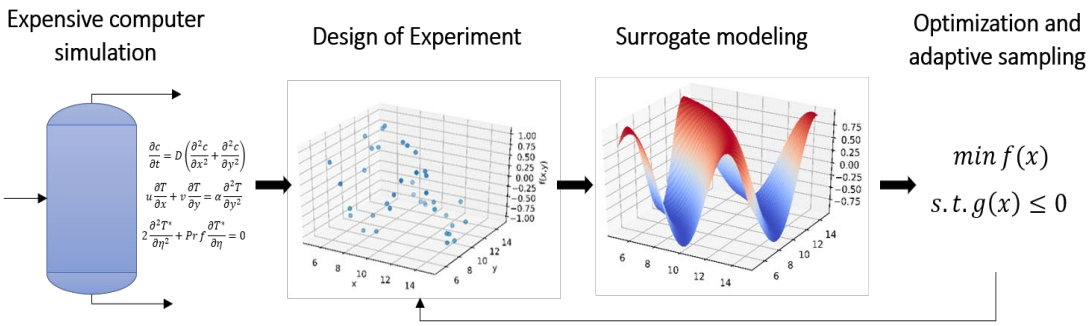


Figure 2. General framework of adaptive surrogate-based optimization. Surrogate models approximate the data from expensive black-box simulations and are iteratively optimized and updated to find an optimal solution

The next step to surrogate-based optimization is constructing a surrogate model, which is a low-fidelity approximation of the actual first-principle model. Currently, various surrogate models exist, and some commonly used surrogate models are shown in Table 1. The selection and parameter fitting of a surrogate model are coupled with a k-fold cross-validation procedure to prevent overfitting. This procedure ensures that the surrogate model parameters are trained only on a subset of samples (training set), while the other subset of remaining samples is left out for validation (validation set). A prediction error on the validation set is calculated using the optimal model. The procedure is repeated k times to allow the selection of the best model with the minimum cross-validation error.

Once a surrogate model is trained using input-output data, analytical representations of the constraints and the objective of a black-box problem become available, which are computationally cheaper to evaluate and compute derivatives for[5]. Several approaches have been proposed for handling constraints, such as fitting a separate model for each constraint[1] or a grouped penalty function[63,64]. In this work, surrogate models are constructed for all black-box constraints and the objective function. A surrogate function can then be optimized by any deterministic optimization solver of choice.

Table 1. Different types of surrogate functions and their function form

Model	Interpolating?	Functional Form	Parameters
Linear	No	$y(x) = b_0 + \sum_{i=1}^{p} b_i x_i$	b_0, b_i for $i = 1, \dots, C$
General Quadratic	No	$y(x) = b_0 + \sum_{i=1}^{p} bl_i x_i + \sum_{i=1}^{p} \sum_{j=1, j \geq i}^{p} bnl_{i,j} x_i x_j$	b_0, bl_i for $i = 1, \dots, C$
Gaussian Process	Yes	$y(x) = \mu + \sum_{n=1}^{N} c_n exp\left[-\sum_{i=1}^{C} \theta_i \left(x_i - x_i^{(n)}\right)^2\right]$	μ, θ_i for $i = 1, \dots, C$ $c_n = f(\theta_i, \mu, \boldsymbol{x}, y)$
Radial basis functions	Yes	$y(x) = \sum_{i=1}^{m} \alpha_i \pi_i(x) + \sum_{n=1}^{N} b_n \varphi(x - x_n)$	α_i for $i = 1, \dots, m$ b_n for $n = 1, \dots, N$

However, regardless of the type of surrogate model used, it is highly unlikely that a perfect model is obtained in just one stage. Hence, surrogate modeling is usually coupled with adaptive sampling to determine the location of the next sampling points to update the model. Two adaptive sampling strategies exist: 1) fitting the best approximation, or 2) facilitating the search of optimum. The first approach involves using adaptive sampling to identify the best approximation of a certain unknown correlation. Once the best approximation is found, it is directly optimized in one-stage using derivative-based optimization solvers[2]. The second approach, however, does not seek to generate the best approximation. Instead, adaptive sampling is used to determine the location of new samples in promising regions to improve the solution. Specifically, it seeks to maintain a balance

between diversity in sampling (i.e. exploration) and optimization (i.e. exploitation). As a result, the surrogate model is used only as an intermediate step to guide the search toward better directions. Once the locations of new sample points have been determined, the simulation is re-inquired at these new points. The surrogate model is then updated using the new design space. This entire process repeats until a convergence criterion is reached.

This work employs the latter adaptive sampling approach for surrogate-based optimization. Instead of searching for the best approximation, a surrogate model is simply used as an intermediate guide to converge to the optimum solution. This approach has been shown to be effective in [1,65,66], especially for problems with computationally expensive simulations. Reduced sampling is achieved because rather than focusing on covering the entire feature space to identify the universally best approximation, this paradigm focuses on areas where the global optimum is likely to be located.

2.3 Surrogate Models

Previously, we have presented a general overview of a surrogate-based optimization algorithm. In this section, we present an in-depth discussion of two specific surrogate models: Artificial Neural Network (ANN) and Gaussian Process (GP). These models are used throughout this work.

2.3.1 Artificial Neural Network Modeling

An Artificial Neural Network (ANN) is a nonlinear statistical model that has been used for both classification and regression [43,67]. Following a standard ANN architecture, the input variable nodes represent the input layer, and the response variable nodes represent

the output layer. The input and output layers are connected by hidden layers. The mathematical expression of an ANN with a single input node (x) and a single hidden layer can be expressed as follows:

$$\hat{f}_{NN}(\boldsymbol{x}) = \sigma\left(\sum_l W_l^{(1)}\,\sigma\left(\sum_h W_h^{(0)} x + b^0\right) + b^{(1)}\right) \tag{2}$$

where h and l represent the number of nodes in hidden and output layers, respectively. The function σ is an activation function, which transfers the input of a node to an output, and $W^{(n)}$ and $b^{(n)}$ are the weights and biases for input-hidden ($W^{(0)}$ and $b^{(0)}$) and hidden-output ($W^{(1)}$ and $b^{(1)}$) layers. The functional form of ANNs depends on the activation function, the number of hidden layers, and the number of nodes in each layer. One commonly used activation function is the hyperbolic tangent function ($\sigma(x) = tanh\,(x)$), while others have been proposed, including sigmoid and ReLU functions [67]. For the final layer, an identity activation function ($\sigma(x) = x$) is used for regression problems [21]. One challenge in constructing a neural network model is hyperparameter optimization. Hyperparameters are variables that determine the structure and training of the network (e.g., number of hidden nodes, number of hidden layers), and these must be set before optimizing the weights and bias values of the neural network. Several strategies can be used to find the optimal hyperparameters, such as grid search, stochastic optimization using a genetic algorithm, and heuristics [67]. Depending on the desired accuracy of the ANN, we can choose different hyperparameter optimization strategies. After the optimal hyperparameters are determined, we can then optimize the weights and bias values of the neural network.

2.3.2 Gaussian Process Modeling

Gaussian process (GP), also known as Kriging, is an interpolating function that assumes that two points that are close to each other are likely to be correlated. This relationship is expressed using a correlation function, which depends on the distance between two points x_j and x_k:

$$cor\left(\epsilon(x_j), \epsilon(x_k)\right) = \exp\left[-\sum_{i=1}^{d} \theta_i \left(x_j - x_k\right)^2\right] \quad (3)$$

The correlation function in (3) captures the following: when two points are close to each other (i.e., distance is small), the correlation approaches to one (high correlation), while when the distance between two points is large, the correlation approaches zero. Using this correlation function, we can obtain the final functional form of GP models:

$$\hat{f}_{GP}(\boldsymbol{x}) = \mu + \sum_{n=1}^{N} c_n \exp\left[-\sum_{i=1}^{k} \theta_i \left(x_i - x_i^{(n)}\right)^2\right] \quad (4)$$

where N represents the number of data points used to train the model, k represents the dimension of the problem, θ_i and c_n represent correlation parameters, and μ is the estimated mean. The parameters can be found by using maximum likelihood estimation (MLE) [65]. The final functional form is directly correlated with the number of data points used to train the model. Hence, the complexity of the functional form increases as more data points are used to construct the model[65].

CHAPTER 3. CONSTRUCTION OF LOW-COMPLEXITY SURROGATE MODELS USING MACHINE LEARNING SUBSET SELECTION FOR REGRESSION

3.1 Introduction

Existing surrogate models can be divided into two broad categories: non-interpolating and interpolating[69]. Non-interpolating models, such as linear, quadratic, polynomial, or generalized regression[70], minimize the sum of squared errors between some predetermined functional form and the sampled data points. While these may lead to simple and interpretable functional forms, they may not be flexible enough to sufficiently capture highly nonlinear correlations [71]. Alternatively, interpolating methods, such as Gaussian Process[71] and radial basis functions (RBF)[5,7,32], exhibit increased flexibility by incorporating different basis functions (or kernels), which are built to exactly predict the training points[69]. Due to model flexibility and accuracy, they have been successfully applied in many areas, such as steady-state flowsheet simulation[72], modeling of pharmaceutical processes[33,73], and aerodynamic design problems[74], to name a few. However, these models tend to have an increased number of parameters and nonlinear, nonconvex terms that are difficult to optimize globally[1].

One promising approach to overcome the aforementioned limitations of existing surrogate-based optimization is Subset Selection for Regression (SSR)[2]. Also known as a sparse representation or sparse coding, SSR involves selecting the most informative subset from a large set of variables or features, which are then linearly combined to generate a

final model. This approach can lower the computational cost by reducing the number of variables or predictors and increase the prediction accuracy by eliminating uninformative features[75]. SSR has been widely applied in numerous fields, such as signal processing[76], gene selection for cancer classification[77], text classification[78], and face recognition[79], but only recently has it been considered for surrogate modeling for optimization[2,80,81]

In this chapter, we explore various SSR techniques for surrogate modeling for optimization. A large set of basis functions or "features" is first created, and a subset of those features is selected to generate a sparse model. This allows us to obtain a low-complexity surrogate model that is computationally cheaper and easier to optimize. Several existing SSR algorithms are compared and tested on both unconstrained and constrained benchmark problems. The novel aspects of this work are: 1) the comprehensive comparison of the performance of various SSR techniques for surrogate modeling with traditionally used interpolating techniques, and 2) the integration of linear SSR techniques for building nonlinear surrogate functions within a surrogate-based optimization framework.

3.2 Motivating Example

In this section, we present a motivating example to introduce the two main concepts of this work. We first illustrate SSR for surrogate modeling. Subsequently, the optimization-based adaptive sampling technique will be introduced. By combining these two strategies, we aim to locate the global optimum with minimum sampling requirements and computation cost.

3.2.1 Surrogate Model Construction using SSR

Data-driven optimization is challenging due to the lack of algebraic expressions of the objective and/or constraints. In this work, we seek to maximize model accuracy and minimize model complexity by using SSR. To use subset selection, we first need to generate a set of features. Conventionally, SSR is used when a large set of original variables exists. A subset of important variables is then selected by removing redundant variables. In this work, we expand the original variable set through nonlinear transformations to obtain a diverse feature set and model complex nonlinear behaviors. For a two-dimensional problem with input variables x_1 and x_2, we can generate a feature set, such as x_1, x_2, x_1^2, x_2^2, $exp\ (x_1)$, and $exp(x_2)$. We can then use SSR to choose only a subset of these features and construct a surrogate model (Figure 3)[2].

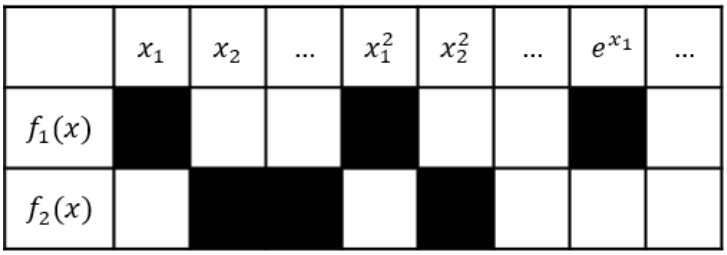
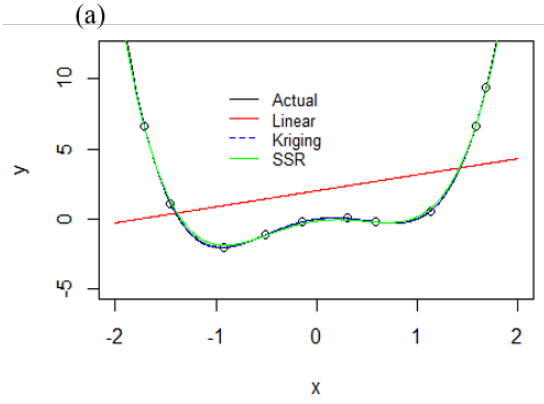


Figure 3. Illustration of SSR in surrogate modeling

We will illustrate this idea by using a simple test function: $f(x) = 2x^4 - 3x^2 + x$. The actual functional form of this test problem is assumed to be unknown, and 10 initial data points are collected by Latin Hypercube Design (LHD). Three different methods – linear, GP, and SSR – are used, and the initial basis set for SSR consists of

x^5, x^4, x^3, x^2, x, and $exp\ (x)$. The resulting surrogate models and their functional forms are shown in Figure 4.

(a)



(b)

Type	Functional Form
Actual	$f(x) = 2x^4 - 3x^2 + x$
Linear	$y = 1.144x + 2.016$
Kriging	$y = 51.989 + \sum_{n=1}^{10} c_n \exp[-3.041\left(x_i - x_i^{(n)}\right)^2]$
SSR	$y = 1.845x^4 - 2.535x^2 + 0.988x - 0.151$

Figure 4. (a) Graphical representation of test problem $f(x) = 2x^4 - 3x^2 + x$, and (b) the resulting functional forms of surrogate models fitted by linear, GP, and SSR

As expected, linear regression leads to an inaccurate model, because it cannot capture the nonlinearity of the actual function. While GP constructs a highly accurate model, its functional form is complicated (i.e., the number of terms is equal to the number of samples used). A balance between model accuracy and interpretability is achieved by SSR, which selects only a subset of basis functions. As a result, SSR can generate a surrogate model with a functional form almost identical to that of the actual test function.

3.2.2 Optimization-Based Adaptive Sampling

One of the most important challenges of surrogate-based optimization is locating the global optimum, while simultaneously minimizing sampling requirements. When data collection is computationally expensive or time-consuming, data points should be collected such that the total number of samples collected at each iteration is minimized, while maximizing the information gained from these sampled points[82]. Since the ultimate goal of

surrogate modeling for data-driven optimization is to locate the global optimum, points sampled near the optimum provide more valuable information than points collected far from the actual optimum. Furthermore, generating a surrogate model that fits all points perfectly, while ideal, would be an inefficient strategy, especially when the number of available samples is limited.

Motivated by this idea, we use optimization-based adaptive sampling in this work (Figure 5). First, an initial sample set is generated using a space-filling experimental design (Latin Hypercube Design), and the first surrogate model is constructed. This initial surrogate model is optimized to global optimality using BARON[83], and the incumbent solutions (i.e., both local and global solutions) are then used as the next sampling point(s). The initial surrogate model is then updated, and this process repeats until a convergence criterion is satisfied. This efficient sampling strategy has been shown to efficiently focus on areas that are more likely to contain the global minimum with high probability. It is important to emphasize that while we are interested in generating a good surrogate model that can accurately predict the optimum, we are not interested in constructing a "perfect" surrogate model that fits all the points exactly. Therefore, as shown in Figure 5, the final surrogate model is allowed to be imperfect in regions of low interest (i.e. near the boundary or areas far from the minimum) but still accurately locates a global minimum.

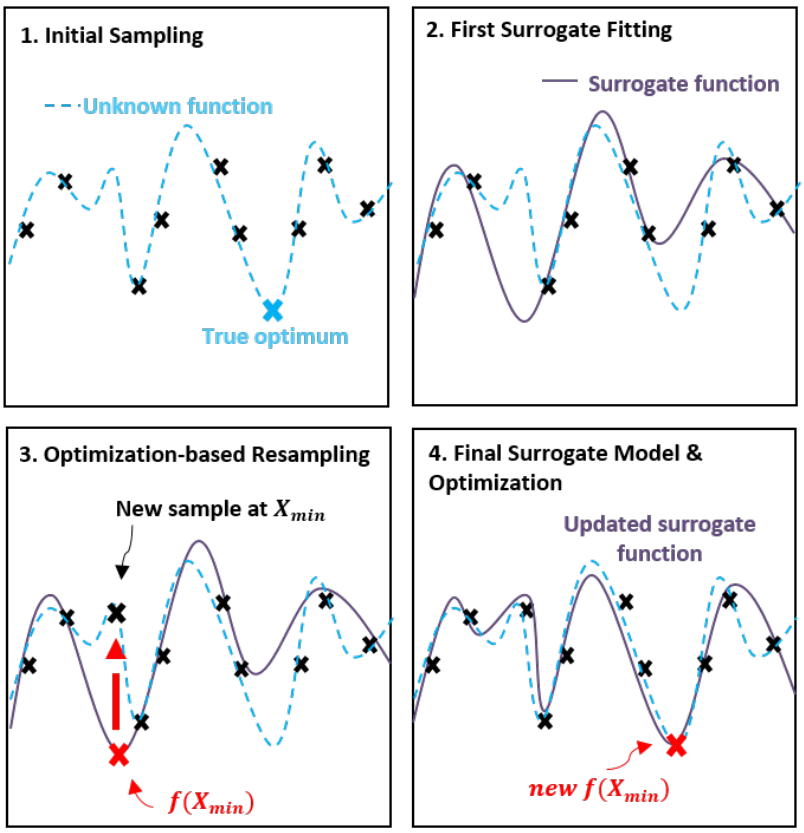


Figure 5. Optimization-based adaptive sampling used in this work

3.3 Subset Selection for Regression

In a typical generalized linear regression setting, we are given a set of training data and we want to generate the following model:

$$y = \beta_0 + \beta_1 X_1 + \cdots + \beta_P X_P = \beta_0 + \sum_{p=1}^{P} \beta_p X_p \, , p = 1, \ldots, P \tag{5}$$

where $X = (X_1, \ldots, X_P)$ is a P-dimensional vector of features or predictors, $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_P)$ is the vector of regression coefficients, y is a response variable. Following an ordinary least squares (OLS) approach, one would minimize the residual sum of squares of

(1) using all of the P features. This is often not desirable, because it is prone to overfitting and does not eliminate redundant features. Therefore, the model interpretability and accuracy can be improved if we locate the best and the sparsest model, which includes only the predictors that explain the true variance of the problem [84].

In this chapter, X_p are the basis functions or the "features" unless mentioned otherwise. These are both linear and non-linear algebraic terms generated by transformations of original variables. Each problem consists of P number of features and n number of samples, and we want to choose only a small subset of the features, specifically q features ($q \leq P$), to create an accurate and sparse model.

One way to achieve this is SSR. SSR, also known as feature selection, can be achieved by either 1) convex optimization or 2) greedy approach. Traditionally, SSR is used to create sparse models, overcome the risk of overfitting, and improve model interpretability[85]. In this work, our main motivation is to investigate whether existing SSR techniques can be embedded within n adaptive sampling surrogate-based optimization framework. We want to create simple and tractable surrogate functions that retain accuracy, yet remain easily optimizable by existing solvers. In addition, we want to compare the performance of SSR-based surrogate functions with that of commonly used complex interpolating functions with respect to computational cost and accuracy in locating the global optimal solution.

3.3.1 Subset Selection using Convex Optimization

A subset selection problem, when formulated as a convex optimization problem, leads to a one-step feature selection. This approach usually involves solving an

optimization problem in the presence of a constraint that leads to a sparse model. In particular, imposing an *L1*-norm constraint has become a popular approach for automatic feature selection[85]. Since the resulting optimization problem is convex, it can be efficiently solved by several optimization solvers. In this section, N represents the total number of data points ($n = 1, .., N$); $\widehat{\boldsymbol{X}}$ is a $[N \times P]$ matrix of features created from the original input variables x_i sampled at N points; $\widehat{\boldsymbol{y}}$ is the vector of the response collected at the samples.

3.3.1.1 Lasso and Elastic Net

One of the most commonly used types of a sparse generalized linear model is Lasso. Lasso (Least Absolute Shrinkage and Selection Operator), originally introduced in [86], is based on the idea of using an *L1*-penalty of the regression coefficient. A sparse linear model is generated by solving the following optimization problem, where λ is a regularization parameter:

$$\underset{\beta \in \mathbb{R}^P}{minimize} \{\left\|\widehat{\boldsymbol{y}} - \widehat{\boldsymbol{X}}\boldsymbol{\beta}\right\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1\} \quad (6)$$

Due to the presence of the *L1*-penalty term, Lasso can perform automatic variable selection by shrinking some coefficients completely to zero. Therefore, it is advantageous over Ridge regression, which minimizes the *L2* penalty on the regression coefficient, because Ridge regression only shrinks the coefficients toward zero but does not enforce them to be equal to zero[87]. While Lasso has been successfully applied in many cases, it has some drawbacks. First, when the number of variables is greater than the number of samples (i.e. $P > N$ case), Lasso can only select at most N variables. Furthermore, if the variables

are highly correlated, then Lasso tends to only select one variable from a set of strongly correlated variables and ignores the grouping effect.

Elastic Net overcomes these limitations by combining *L2* and *L1* norm penalty terms. The presence of an *L2* norm makes Elastic Net penalty strictly convex for all $\alpha > 0$, and this strict convexity guarantees a grouping effect and overcomes the limitation on the number of variables selected in the $P > N$ case. Hence, a group of highly correlated predictors has approximately the same coefficients, whereas Lasso only selects one of the predictors due to its non-strictly convex penalty term[87]. The generalized form of the Elastic Net optimization problem is as follows[88]:

$$\underset{\beta \in \mathbb{R}^P}{minimize} \left\{ \frac{1}{N} \left\| \hat{\boldsymbol{y}} - \hat{\boldsymbol{X}}\boldsymbol{\beta} \right\|_2^2 + \lambda \left[\alpha \|\boldsymbol{\beta}\|_1 + \frac{1}{2}(1-\alpha)\|\boldsymbol{\beta}\|_2^2 \right] \right\} \tag{7}$$

The last two terms are the Elastic Net penalty, which is a combination of both Lasso and Ridge penalties. The Elastic Net penalty is controlled by α: if $\alpha = 1$, the Elastic Net penalty is equivalent to Lasso regression; if $\alpha = 0$, it becomes a simple Ridge regression. The tuning parameter λ controls the overall strength of the penalty and the degree of regularization.

3.3.1.2 Sparse Principal Component Regression

Principal component regression (PCR) is a two-stage procedure that performs principal component analysis (PCA) followed by ordinary least squares (OLS) regression. In particular, the regression is performed by using principal components as new explanatory variables instead of original variables, and the accuracy of the model can be controlled by varying the number of principal components included in the model[89].

However, since the principal components are linear combinations of all original variables, feature selection is not directly feasible via PCA. Consequently, a sparse model cannot be created solely by PCR.

To overcome this limitation, Zou et al.[89] proposed a new method called sparse principal component analysis (SPCA), which imposes the Elastic Net penalty on the regression coefficients. This method generates principal components with sparse loadings, which can be combined with OLS to create a sparse regression model. The sparsity of principal components is controlled by the Elastic Net penalty λ. This method will be noted as "sPCR1" in this work. However, one disadvantage of sPCR1 is that it is an unsupervised learning technique. The principal components are selected without utilizing information on the response variable, which could potentially degrade the performance of the regression model.

Kawano et al. proposed a supervised, one-stage approach for principal component regression in [90], which performs sparse PCA and regression simultaneously. It obtains sparse principal components that are related to the response variable. The model sparsity is obtained by imposing a penalty onto the regression coefficients using two regularization parameters (λ_β and λ_γ). This method will be referred as "sPCR2".

3.3.1.3 Sparse Partial Least Squares Regression

Partial least squares (PLS) regression has been widely used as an alternative to OLS and PCR because of its robustness. Specifically, it has been found that model parameters do not drastically change as new samples are taken from the total population[91]. Unlike PCA, PLS is a supervised dimensionality reduction technique. However, similar to PCA,

PLS does not lead to automatic feature selection as the final direction vectors are linear combinations of all of the original predictors. To address this problem, Chun et al.[92] have proposed a sparse Partial Least Squares regression (sPLS). This method imposes an *L1*-constraint on the dimension reduction stage of PLS and constructs a regression model by using only a subset of sPLS components as new explanatory variables [92]. Note that even though both sPLS and sPCR2 are supervised dimensionality reduction techniques, they have different loss functions: sPLS seeks to maximize the covariance between input X and output Y while imposing an elastic net penalty onto a surrogate of the direction vector; sPCR2 has both the regression loss and the PCA loss functions with sparse regularization on the regression coefficients[90,92].

3.3.2 Subset Selection using a Greedy Approach

Greedy algorithms involve iteratively adding or removing features, which can be achieved by either forward selection or backward elimination. In forward selection, a model is generated by progressively incorporating variables and generating a larger subset; in backward elimination, the model starts with all predictors and iteratively eliminates the least promising ones. Once the models are trained, the most or least predictive features can be chosen by criteria that depend on measures such as the root mean squared error (RMSE), the cross-validation error, or the value of model coefficients (weights)[93]. In this work, Support Vector Regression (SVR) is chosen as a regression strategy. Only backward elimination is considered in this work, because forward selection has been reported to find weaker subsets[93]. This behavior can be explained since the importance of variables is not assessed in the presence of other variables that are not included.

3.3.2.1 Support Vector Machine – Recursive Feature Elimination (SVM-RFE)

Support vector machines (SVM) seek to find a regularized function $f(\boldsymbol{X})$ that separates the data with at most ε deviation from the actually observed data y_n, while making sure $f(\boldsymbol{X})$ is as flat as possible. If a data set is "linearly separable" (i.e., a linear function can separate a set of data without error), $f(\boldsymbol{X})$ takes the form:

$$f(\boldsymbol{X}) = \boldsymbol{w} \cdot \boldsymbol{X} + b, \qquad \boldsymbol{w} \in \mathbb{R}^P, b \in \mathbb{R} \tag{8}$$

where $\boldsymbol{w}$ is a P-dimensional weight vector and b is a bias value. In this work, we train a linear SVR model; however the final functional form is nonlinear because we use the augmented set of nonlinear features ($\boldsymbol{X}$).

The values of $\boldsymbol{w}$ and b can be found by solving the following optimization problem[94]:

$$\begin{aligned} &minimize \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}(\xi_n + \xi_n^*) \\ &subject\ to \begin{cases} y_n - \langle \boldsymbol{w}, \widehat{\boldsymbol{X}}_n \rangle - b \leq \varepsilon + \xi_n & n = 1, .., N \\ \langle \boldsymbol{w}, \widehat{\boldsymbol{X}}_n \rangle + b - y_n \leq \varepsilon + \xi_n^* & n = 1, \dots, N \\ \xi_n, \xi_n^* \geq 0 & n = 1, \dots, N \end{cases} \end{aligned} \tag{9}$$

This formulation generates a model that is a linear combination of all original features; thus, none of the original input features can be discarded. To generate a sparse model using SVM, Guyon et al.[77] developed a pruning technique that eliminates some of the original features and generates a subset of features that yields the best performance.

Recursive feature elimination (RFE) is iteratively used to rank the features and remove less important features by performing three simple steps: 1) train an SVM model and obtain the weight vector $\boldsymbol{w}$, 2) compute the ranking criterion (w_i^2) for all features, and 3) remove the feature with the smallest ranking criterion. The squared weights w_i^2 are used as a ranking criterion, because the magnitude of w_i^2 denotes the importance of a feature to the overall model.

When the number of original features is large, it is computationally inefficient to remove a single feature per iteration. Instead, several features can be removed at each iteration, but this has to be cleverly done to not sacrifice performance accuracy. In this case, the method provides a feature subset ranking instead of a feature ranking, such that $F_1 \subset F_2 \subset \cdots \subset F$. Hence, the features in a subset F_m should be taken together to generate a model. In this work, we employ an adaptive multiple-feature removal strategy. Initially, when the number of remaining features is large, more features are removed to speed up the elimination process. When only a few features remain, fewer features are removed to more carefully explore synergistic effects between remaining features. This is achieved by using a heuristic rule that we have found efficient, which removes $1/(iter + 4)$ features at a time, where $iter$ represents the iteration number. Therefore, when $iter = 1$, one-fifth of the existing features are removed, whereas when $iter = 10$, fewer features (1/11 of remaining features) are removed.

The performance of SVM depends on the selection of two hyper-parameters: C and ε. The cost parameter C is a regularization term that represents the cost of constraints violation, which controls how much samples inside the margin contribute to the overall

error. Furthermore, ε defines a margin of tolerance, in which no error penalty is given to a point lying inside this margin. These two parameters in conjunction control the width and the flatness of the margin, and they are usually tuned by performing a grid search or cross-validation[94]. However, these commonly used approaches can be computationally expensive. Therefore, we instead use the following equation to obtain C directly from the training data[95]:

$$C = max\left(\left|\bar{y} + 3\sigma_y\right|, \left|\bar{y} - 3\sigma_y\right|\right) \tag{10}$$

where $\bar{y}$ and σ_y are the mean and standard deviation of the y values in the training set. After the model is trained, the features are ranked and removed according to the value of w_i^2 until the change in RMSE of two consecutive iterations is greater than 10%.

3.4 Overall Algorithm

We assess the performance of five SSR techniques by integrating them into a framework that has been developed to optimize black-box simulation-based problems, ARGONAUT[1]. ARGONAUT involves subcomponents to perform sampling, surrogate function construction, global optimization of the surrogate-based formulations, and optimization-based adaptive sampling to accurately locate the global minimum[1,13].

3.4.1 Basis Function Generation

One advantage of SSR in surrogate modeling is that simple yet diverse and flexible basis functions can be used instead of highly complicated Gaussian or radial basis functions (Table 2). The original variable set is transformed by simple nonlinear transformations,

such as polynomial and multinomial transformations. Other types of basis functions can be easily integrated, if needed. More complicated terms, such as trigonometric functions, are not directly handled by most optimization solvers; thus, they were not included in this work. The basis types that are included in the initial superset of features are shown in Table 2, where $\tau = \{1,2,3,4\}$, $\alpha = \{1,2,3\}$, $\beta = \{1,2,3\}$, $\gamma = 1$, $\eta = 1$, and indices i, i', i'' represent different variables of the original input space.

Another noteworthy advantage of SSR is that *a-priori* knowledge of the system can be used to improve model performance. If the actual functional form is fully or partially known based on first-principles or heuristics, the user can selectively include or exclude certain basis functions. For example, if we want to generate a surrogate model for a second-order reaction (rate $= \mathrm{k[A]^2}$), we can include x^2 in the basis set to guide the selection of the basis function toward the term included in the actual rate equation. For a greedy approach (i.e., SVM-RFE), it is even possible to guarantee the inclusion of this term in the final model by assigning an arbitrarily large weight to x^2 at every iteration.

Table 2. List of possible basis functions

Type	Equation
Polynomial	x_i^{τ}
Multinomial	$x_i^{\alpha} \cdot x_{i'}^{\beta}$ and $x_i \cdot x_{i'} \cdot x_{i'''}$
Exponential	$exp\left(\frac{x_i}{\gamma}\right)^{\eta}$
Logarithmic	$ln\left(\frac{x_i}{\gamma}\right)^{\eta}$

3.4.2 Surrogate Model Construction

A surrogate model is generated by the selected SSR method or by fitting an interpolating function. All SSR methods require tuning of hyper-parameters for optimal performance, and the specifics of how each method is tuned can be found in Table 3. The initial sample set is divided into k training and k validation sets, and k models are generated using all SSR methods. The best model is determined by calculating the root-mean-square error (RMSE) of k models on the validation set. The model with the smallest RMSE is chosen to proceed to the next stage. Similarly, the constraints are modeled with the selected method.

3.4.3 Optimization-Based Adaptive Sampling

Latin Hypercube Design is used for initial sampling. Based on traditionally used heuristics[1,13], when the dimension of the problem is less than or equal to 20, $10M + 1$ samples are used; when the dimension is greater than 20, a fixed number of 251 samples are collected. The simulation is inquired at these points, and the computation cost of sampling is reduced by parallelizing the procedure computationally. After initial sampling, optimization-based adaptive sampling is used to update the surrogate models. The initial surrogate model is globally and locally optimized by using solvers BARON[83] and CONOPT[103], respectively. This allows us to find the global solution and a diverse set of local solutions. The simulation is then re-inquired at all of these incumbent solutions. The algorithm terminates if one of the following convergence criteria is met: 1) the incumbent solution does not improve over a consecutive set of iterations, 2) the maximum number of function calls has been reached, and 3) a feasible incumbent solution is found with a very low cross-validation RMSE [1].

Table 3. Algorithm parameters

Method	R package	Parameters	Method of tuning
Elastic net	glmnet (`glmnet`) [96*]	α = controls Elastic Net penalty λ = controls the overall strength of the penalty	Grid search using 10-fold cross validation
sPCR1	elasticnet (`spca`) [97]	K_{spcr1} = number of components $sparse$ = controls the number of sparse loadings $para$ = vector of 1-norm penalty parameter	K_{spcr1} = number of basis functions $sparse$ = "penalty" $para$ = 0.01
sPCR2	spcr (`cv.spcr`, `spcr`) [98]	λ_β = nonnegative regularization parameters for regression coefficients λ_γ = nonnegative regularization parameter for regression intercepts	Grid search using cross validation
sPLR	spls (`cv.spls`, `spls`) [99]	η = thresholding parameter K = number of hidden components	η = 10-fold cross-validation K = range(1, $min\{p, 0.9n\}$) [100]
SVMRFE	kernlab (`svmLinear`) [101*]	C = controls margin softness ε = margin of error tolerance	$C = max(\lvert\bar{y} + 3\sigma_y\rvert, \lvert\bar{y} - 3\sigma_y\rvert)$ $\varepsilon = 0.1$

*The selected SSR method is coupled with R package caret [102] via `train` to perform cross-validation/grid search

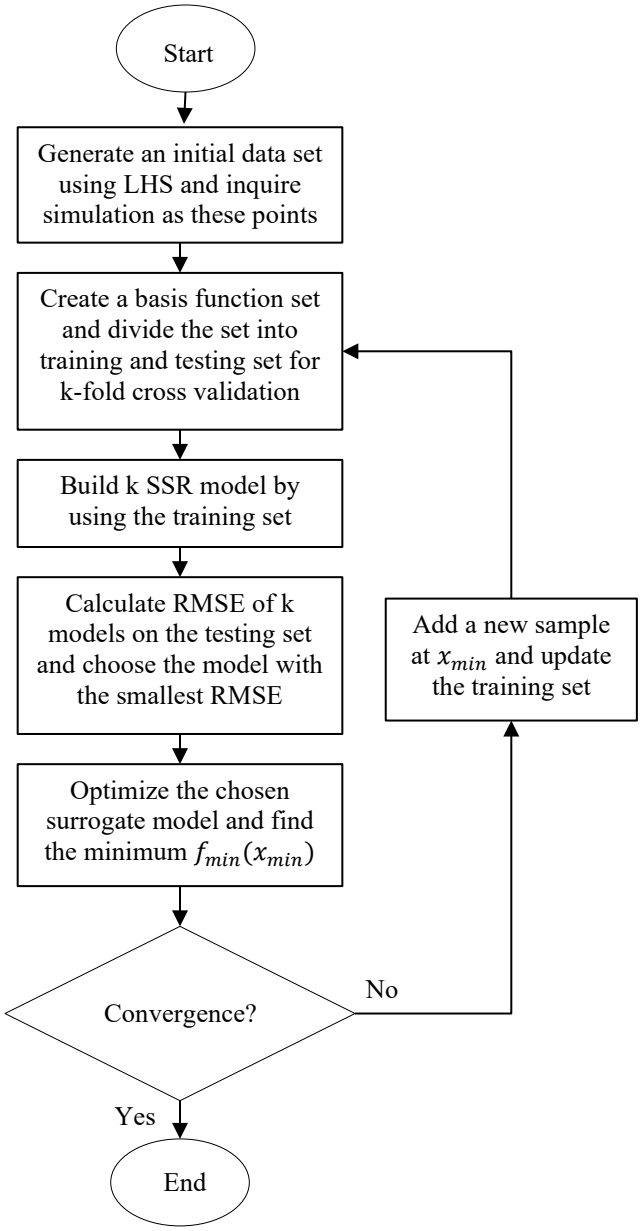


Figure 6. Algorithmic flowchart of the proposed algorithm. At each iteration, an SSR model is constructed and optimized. If no satisfactory solution has been found, new data points are added to the existing data set and the steps are repeated.

3.5 Computational Studies

We test the five aforementioned SSR techniques on two sets of benchmark problems. The performance of SSR is compared to that of a Gaussian Process (GP)

model[65], a widely used interpolating surrogate model introduced in Chapter 2. The best performance out of three runs is chosen to evaluate the performance of all algorithms. All methods are tuned according to Table 3 to find the best hyper-parameters; $k = 5$ is used for cross-validation. Both sets contain problems that are linear, nonlinear, convex, and/or nonconvex, and all problems have known bounds and known global minima. The functions contain a diverse set of algebraic terms, which may or may not be in the generated basis function set shown in Table 1. Furthermore, both the objective and constraints of the problems are assumed to be unknown, following the definition of a black-box problem.

3.5.1 Test Set A: Unconstrained Problems

The first test set consists of a subset of 191 test problems from Sahinidis library[11]. All problems are unconstrained with known bounds. The algorithm is parallelized using 2 processors[1]. The dimension and the number of problems in each set are shown in Table 4.

Table 4. Specifics of test set A

Dimension of problems	Number of problems	Number of basis functions
2	73	19
3-4	48	43-78
5-9	47	125-453
10-16	14	575-1720
20-30	9	2950-8155

Model performance is evaluated by comparing the fraction of problems solved with the number of function calls and the total computation time. The number of function calls is important to evaluate model performance because black-box simulations can be computationally expensive, limiting the number of samples that are collected. The overall computation time is the total CPU time required to perform sampling, parameter

estimation, and surrogate optimization. As our goal is to find a global minimum and not necessarily to find a surrogate model that fits all points perfectly, we evaluate the model performance based on the accuracy of the obtained optimum. Specifically, the error is normalized by the median of all samples so that the range of the search space is taken into account when evaluating the merit of a solution.

The performance profiles of all SSR techniques and GP are shown in Figure 7. The problem is considered to be solved if the normalized error is within 1% ($\varepsilon = 0.01$) of the global solution. As shown in Figure 7, GP solves the most number of problems and shows superior performance to SSR. However, it usually requires the most number of samples to solve the same fraction of problems. This implies that GP might not be a favorable choice when the number of samples is the most important limiting factor. sPLR, while it does not solve as many problems as GP, solves more problems than GP if only up to 500 points are sampled. Furthermore, all other SSR techniques, such as Elastic net, sPCR1, sPCR2, and SVMRFE show relatively similar performance, with Elastic net slightly better than the other three.

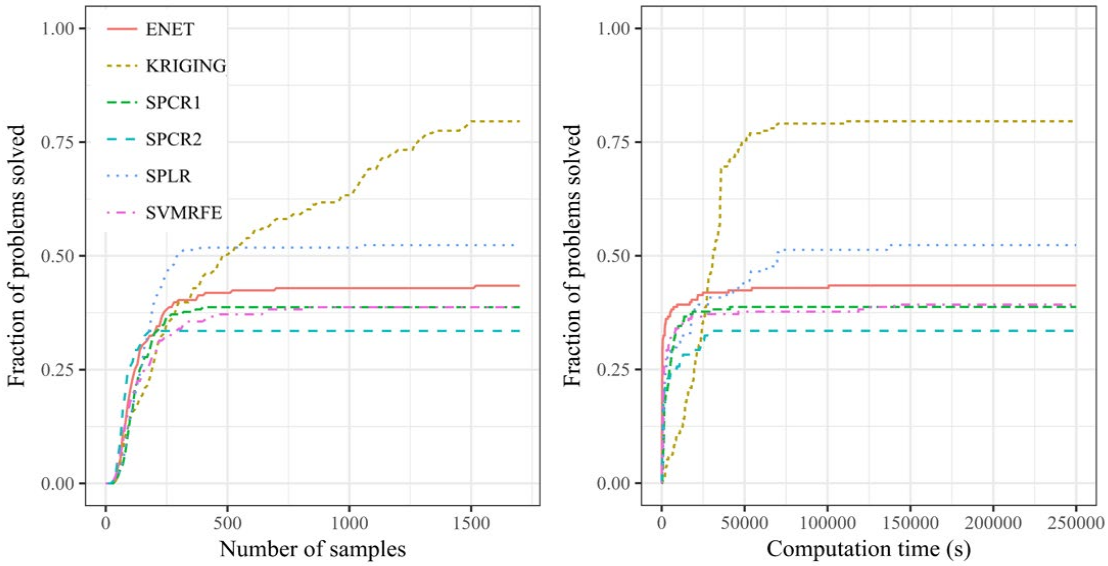


Figure 7. Performance profile of test set A (ε = 0.01) for (a) number of samples and (b) computation time

Next, we test the performance of all algorithms in solving the problem with a higher error tolerance (Figure 8). The purpose of this test is motivated by the nature of data-driven applications, for which a 1% error tolerance might be too strict. For example, the simulation or data may contain uncertainty due to numerical or measurement errors; therefore, locating an optimal solution in the neighborhood of the optimum with the fewest samples possible may be sufficient. When the convergence criterion is relaxed to 10%, the fraction of problems solved increases drastically for all SSR methods. As expected, the performance profile for GP did not change significantly. This suggests that while all SSR methods are good at determining the approximate location of the optima, GP is better at locating a very precise solution. This can be rationalized by the fact that GP is an interpolating method, and hence the model is very flexible to represent highly nonlinear input-output relationships.

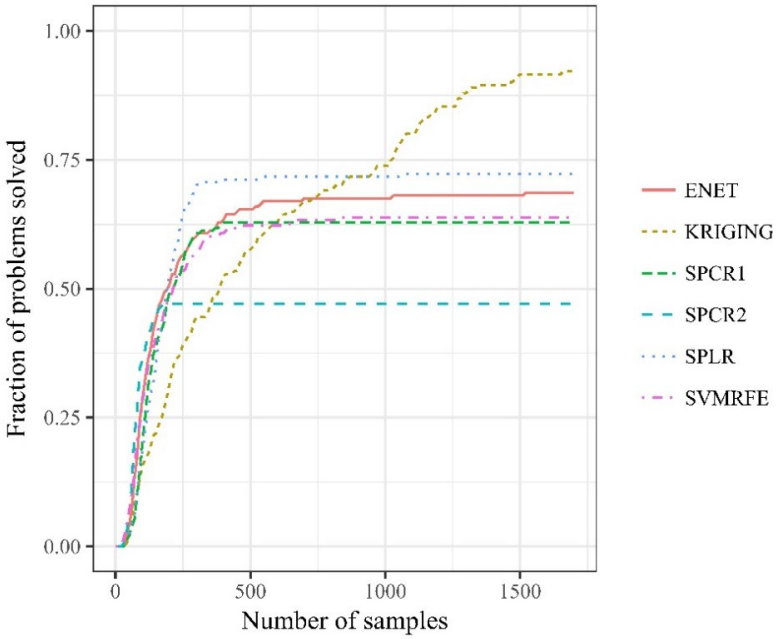


Figure 8. Performance profile of test set A ($\boldsymbol{\varepsilon}$ = 0.1)

3.5.2 Test Set B: Constrained Problems

The second test set is from GlobalLib[1], in which the problems have inequality constraints with known bounds and global minima (Table 5). As these problems have multiple constraints, different surrogate models are fitted for the objective and each of the constraints. This procedure is performed in parallel. For all problems, none of the objective nor the constraints are assumed to be known to test the algorithm in the most challenging black-box case. The model performance is compared by using the same convergence criterion that is used for the test set A. Both $\varepsilon = 0.01$ and $\varepsilon = 0.1$ are used to generate performance profiles.

Similar to test set A result, Figure 9 shows that GP solves the most number of problems (~75%), followed by sPLR, which solved about 55% of the problems. When the error tolerance is increased to 10%, Figure 10 shows that the fraction of problems solved

increases drastically. Therefore, we can conclude that SSR is good at determining the approximate location of the global solution with less samples. Furthermore, we noticed that SPLR and GP exhibit similar performance when the problem dimensionality is low. In fact, for up to 5-dimensional problems, SPLR solves 91% of the problems, and GP solves 95% of the problems. This implies that SSR performs well when the dimensionality is low, but its performance degrades as the problem dimension increases. For high-dimensional problems, SSR-based surrogate models are not flexible enough to very precisely determine the global solution.

Table 5. Specifics of test set B

Dimension of problems	Number of constraints	Number of problems	Number of basis functions
2-3	1-10	31	19-43
4-6	1-12	24	78-185
7-10	4-14	16	259-575
11-30	9-22	17	715-8155

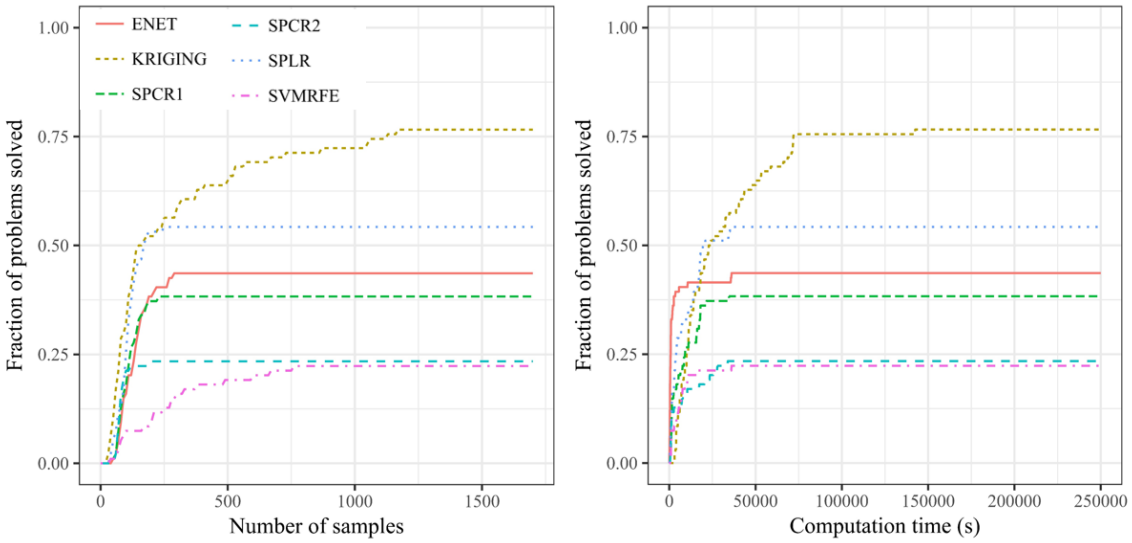


Figure 9. Performance profile of test set B (ε = 0.01) for (a) number of samples and (b) computation time

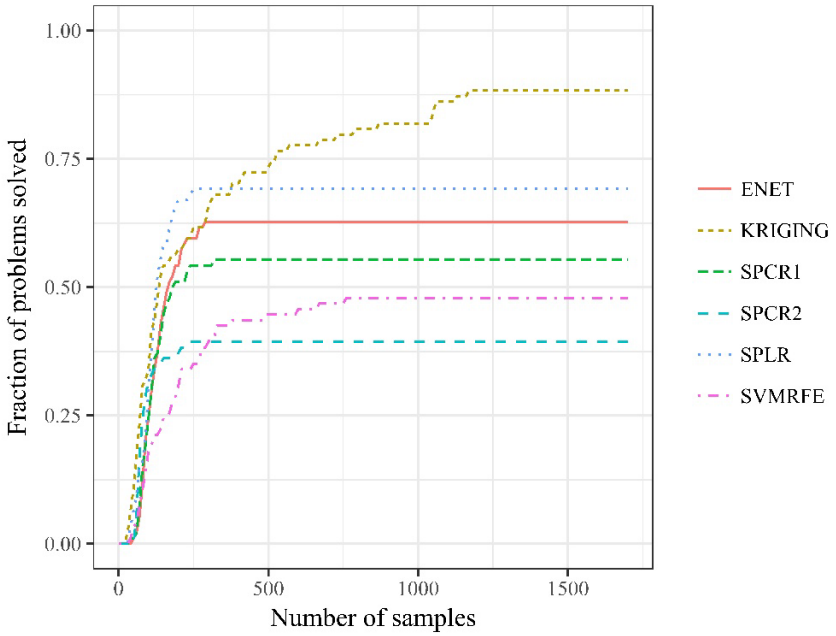


Figure 10. Performance profile of test set B (ε = 0.1)

3.5.3 Computational Time

One advantage of using SSR over GP is that SSR leads to low-complexity models that are easier to globally optimize. Figure 11 shows the breakdown of CPU time for all problems that are solved within 1% error. The sampling stage represents the total computational time to run the simulation and collect the samples for both initial and adaptive sampling. The parameter estimation stage includes the total computation time to perform parameter estimation, cross-validation, and surrogate model construction for all objective and constraints with parallelization. This is usually the most computationally intensive step in the algorithm, especially when the problem involves multiple constraints. Lastly, the optimization stage involves global optimization of the surrogate model.

As expected, GP requires the highest computation time to globally optimize due to the high complexity of the surrogate model (Figure 11c). The computation time required

to optimize a GP model is on average 500 times greater than those of SSRs. However, GP is the most computationally efficient model with respect to the cost required for parameter estimation Figure 11b). Therefore, while GP leads to a more complicated model, this can be compensated by improved accuracy and a less computationally intensive parameter estimation stage of the model.

Lastly, while the computation times required for sampling stage (Figure 11a) and optimization (Figure 11c) exhibit no significant difference between all SSR methods, the computational cost for parameter estimation stage differs significantly (Figure 11b). In general, Elastic Net is the most computationally efficient model, because the model only requires a simple optimization problem to be solved to create a sparse model. SVMRFE, while computationally intensive when only one feature is removed at a time, overcomes this limitation by removing multiple features at a time as previously discussed. All other methods, including sPCR1, sPCR2, and sPLR, exhibit slower parameter estimation performance, especially for high-dimensional problems.

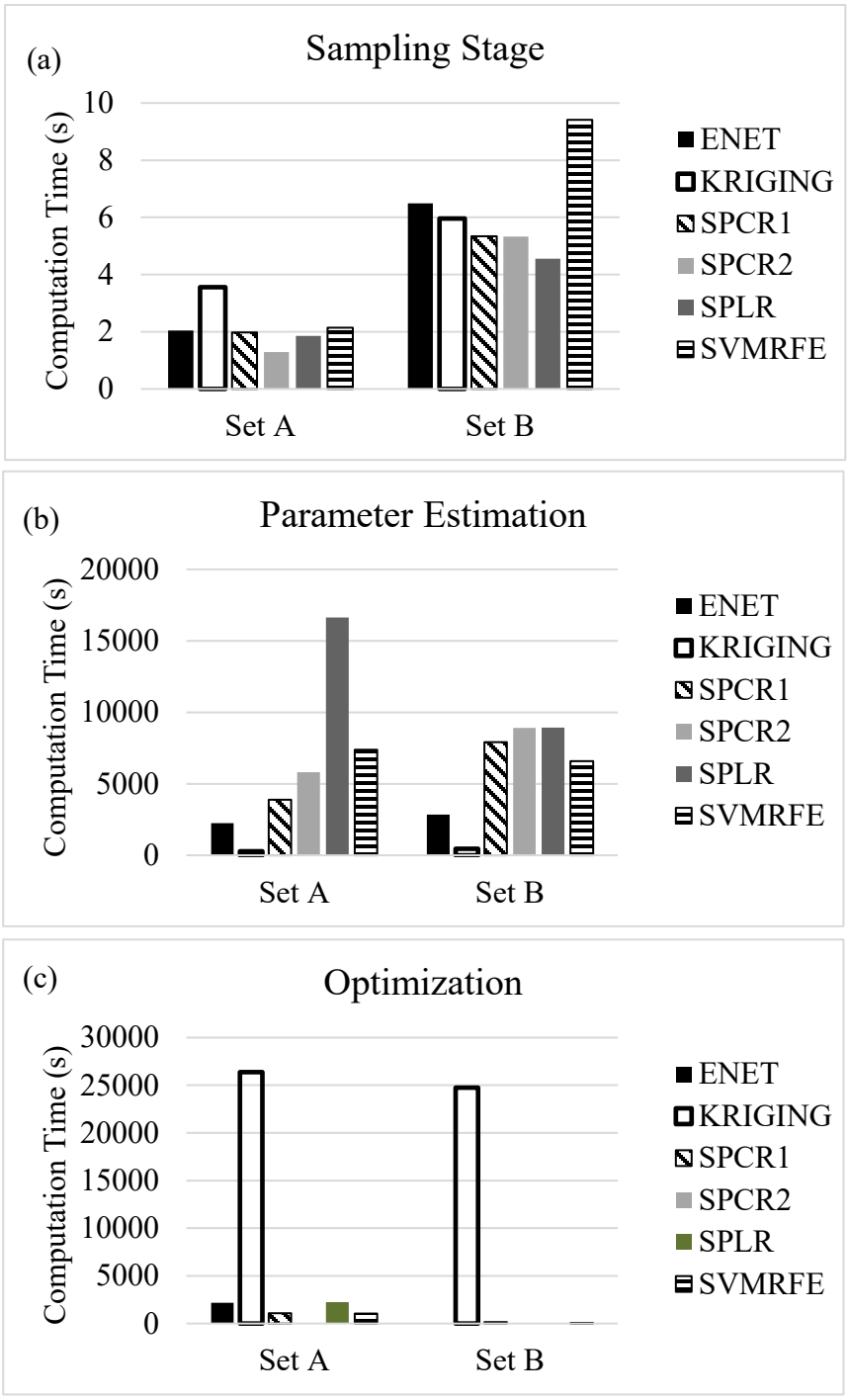


Figure 11. Breakdown of computational cost for all solved problems (ε = 0.01) for (a) sampling stage, (b) parameter estimation stage, and (c) optimization stage

3.5.4 Model Complexity

In this section, we further explore our results with respect to the model complexity of the resulting surrogate models. A method that generates a simpler model with fewer terms is more desirable because a simpler model is generally easier to optimize. In this work, the model complexity is defined by computing the sparsity of the final model:

$$\text{Model sparsity} = \frac{\text{\# of selected basis functions}}{\text{\# of all possible basis functions}} \tag{11}$$

A model with sparsity close to 0 contains very few features, and a model with sparsity 1 contains all original features. The model sparsity of the final surrogate model of the objective is shown for both test set A and B (Figure 12). ENET and SVMRFE usually lead to the sparsest final model, which only retains 20-30% of the original features. sPLR, sPCR1, and sPCR2, on the other hand, keep about 70% of the original features.

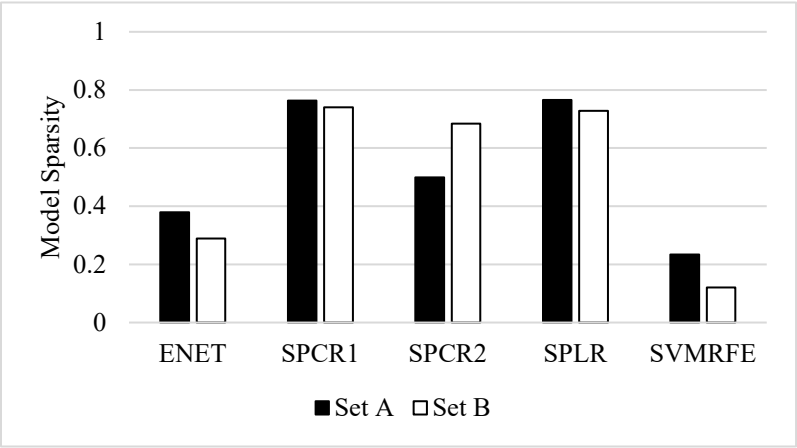


Figure 12. Model sparsity of the objective for all solved problems for ε = 0.01

Model sparsity becomes more important as the dimension of the problem increases. For example, a 20-dimensional problem leads to 2950 initial basis terms. Therefore, if only 10% of the features are eliminated via SSR (model sparsity = 0.9), the resulting model is a linear combination of 2655 terms. Therefore, if we consider both the accuracy of the solution and model complexity, we can conclude that either ENET or sPLR shows the best performance. Since ENET leads to generally sparse solutions and sPLR generally leads to a higher accuracy solution, sPLR is preferable to ENET when the dimension of the problem is low, but ENET can offer a better and faster solution for higher-dimensional problems.

3.6 Conclusions

In this chapter, we present a comprehensive comparison of five different subset selection for regression techniques for surrogate modeling. We investigate the hypothesis of whether subset selection for generalized regression compared to complicated kernel-based interpolating surrogate functions is better for data-driven optimization. Different subset selection methods are tested over a large set of box-constrained and constrained benchmark problems with up to 30 dimensions, and their performance is compared to that of a popular interpolating surrogate modeling technique, GP. While a GP-based approach solves the most number of problems, our results indicate that the computational time required to optimize a complex interpolating model is many orders of magnitude greater than those of SSRs. Nevertheless, GP requires the least computational time for parameter estimation, which overall may justify the higher computational cost for the model optimization. In addition, our results indicate that when using regression surrogate functions, more problems are solved when sampling is very limited. All subset selection methods show promising performance, especially for low-dimensional problems; however,

their performance degrades as the dimension of the problems increases in addition to their high computational cost for selection of features and identification of the model parameters.

Despite some of the good properties of SSR techniques, this chapter shows that the use of more flexible surrogate models (e.g., nonlinear kernel and nonparametric) results in overall better performance. For the subsequent work, we therefore focus on ANN and GP.

CHAPTER 4. SURROGATE-BASED OPTIMIZATION OF MIXED-INTEGER NONLINEAR PROBLEMS

4.1 Introduction

Surrogate-based optimization has been extensively studied for nonlinear optimization problems (NLPs) with continuous input or decision variables[1,2,105]. However, many chemical engineering problems contain both continuous and discrete (integer or binary) decision variables. For example, the design of a distillation column involves continuous variables for operating conditions and discrete variables for the number of stages. Similarly, a superstructure synthesis optimization problem contains binary variables to represent design configurations, while nonlinear relationships represent phenomena within the processes [15,18,106-108]. In [109], a case study on the design of solar plants is discussed, in which the discrete decisions are embedded in the simulation. This leads to a simulation-based optimization problem that cannot be relaxed or decoupled with respect to discrete and continuous variables.

There are a few black-box mixed integer nonlinear programming (bb-MINLP) optimization algorithms proposed in the direct-search literature[53,54,110,111] and in the surrogate-based literature[55,59,60,62], which are described in detail in the next section. Nevertheless, the optimization of bb-MINLPs is still a difficult problem due to several open challenges that are intrinsic to MINLP[62]. The first challenge is obtaining a representative, tractable, and balanced sample set when both discrete and continuous variables are present. When all of the decision variables are continuous, space-filling sample designs (e.g., Latin

hypercube[112], orthogonal arrays[113], and Sobol sequences[114]) are used to generate balanced sample sets, and the simulation is inquired at these points. These sampling methods cannot be directly applied for bb-MINLP problems because the simulation may not provide output values at non-integral values of the discrete variables[55]. Another challenge is the surrogate model fitting in the case of mixed-variable inputs. Existing surrogate modeling algorithms for bb-MINLP [55,59,60,62] assume all variables are continuous in order to obtain a smooth and continuous surrogate model. At the same time, surrogate models assume that all input variables are ordinal, which means that a higher value corresponds to higher intensity, such as temperature or pressure levels. Binary variables do not satisfy this assumption as "0" and "1" usually represent different choices, as opposed to intensity. While this limitation can be overcome by using multiple surrogates and patching them at discontinuities (e.g., piecewise functions), this could complicate the surrogate-based optimization formulation significantly[115].

A mixed-variable response surface introduced in [115] will be used here to demonstrate the aforementioned challenges (Figure 13). This response surface has one continuous variable (x) and one discrete variable (y) with three possible levels $y = [-2,0,2]$. When plotting the response surface for different levels of y, one can observe that the behavior of the output is very different (Figure 13a). There are multiple ways to approximate and subsequently optimize this problem, such as: 1) treat each level of y as an independent problem by fitting and optimizing separate surrogate models, or 2) assume continuity in all variables and fit a single continuous response surface with sparse sampling in the y direction. The first approach is possible in low dimensions but would become intractable as the number of discrete variables and levels increases. On the other hand, if

continuity in all variables is assumed, this could lead to inaccurate surrogate models since there will be no samples in between non-integral values of the discrete variables. For the same example, if this black-box input-output relationship is assumed to be a 2D continuous function (Figure 13b), the complexity of the surface becomes apparent. More specifically, at the middle level of y, the function has a very sudden and steep change in response; thus, assuming continuity when fitting this response surface may lead to inaccurate surrogate models. Most importantly, if the level values of y do not represent an intensity, then the assumption of continuity in y is problematic. In this work, we study various techniques to obtain a representative set of samples and fit appropriate surrogate models for mixed-variable optimization problems. The presence of nonlinear constraints, both inequality and equality, and non-convexity of the problems all pose further challenges for the solution of bb-MINLP problems[9].

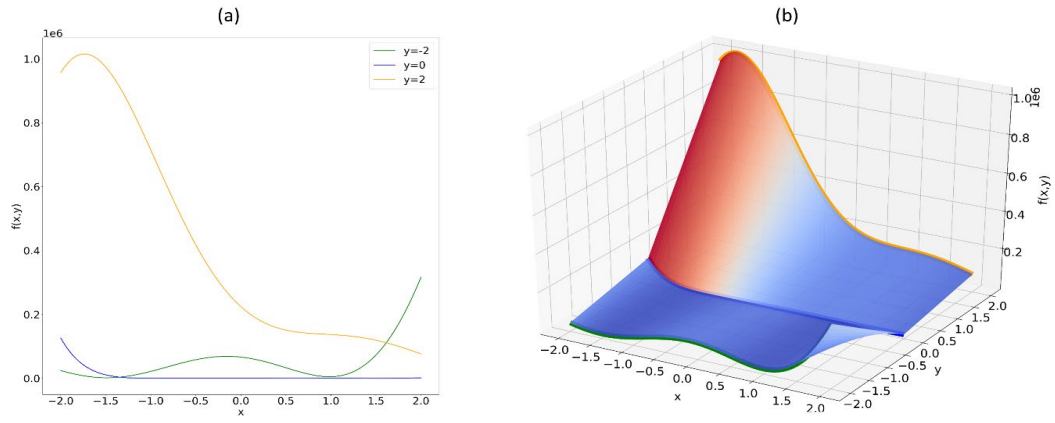


Figure 13. Goldstein price function adapted from [115]. For each value of discrete variable y, the function exhibits a drastically different behavior. In (a) different y level values are plotted separately, in (b) the 2D surface is plotted.

In this chapter, we aim to develop an algorithm to solve the following black-/gray-box MINLP (12):

$$\begin{aligned} &\min f(\boldsymbol{x}, \boldsymbol{y}) \\ &s.t.\ g_{IB}(\boldsymbol{x}, \boldsymbol{y}) \leq \mathbf{0} \\ &g_{IK}(\boldsymbol{x}, \boldsymbol{y}) \leq \mathbf{0} \\ &h_{EB}(\boldsymbol{x}, \boldsymbol{y}) = \mathbf{0} \\ &h_{EK}(\boldsymbol{x}, \boldsymbol{y}) = \mathbf{0} \\ &\boldsymbol{x}^l \leq \boldsymbol{x} \leq \boldsymbol{x}^u, \qquad \boldsymbol{y} \in \{0,1\}^{k_2} \\ &\boldsymbol{x} \in \mathbb{R}^{k_1}, \qquad k = k_1 + k_2 \end{aligned} \tag{12}$$

where $\boldsymbol{x}$ represents continuous variables, $\boldsymbol{y}$ represents binary variables, $\boldsymbol{x}^l$ and $\boldsymbol{x}^u$ represent the lower and upper bounds of the continuous variables, k_1 and k_2 represent the dimensions of continuous and binary variables respectively, $f(\cdot)$ represents black-box objective, $g_{IB}(\cdot)$ and $g_{IK}(\cdot)$ represent inequality constraints that are unknown (black-box) and known, respectively. Similarly, and $h_{EB}(\cdot)$ and $h_{EK}(\cdot)$ represent equality constraints that may be unknown and known, respectively. Sets IB and EB represent the set of black-box inequality and equality constraints, respectively. Similarly, sets IK and EK denote the sets of known inequality and equality constraints, respectively. If all constraints and objective are unknown (i.e.,$IK = \emptyset, EK = \emptyset$), the problem will be referred as a black-box MINLP (bb-MINLP). If some constraints are known, the problem will be referred as a

gray-box MINLP (gb-MINLP). $g_{IK}(\boldsymbol{x}, \boldsymbol{y})$ and $h_{EK}(\boldsymbol{x}, \boldsymbol{y})$ represent the known inequality and equality constraints, and these can be handled directly without constructing surrogate models.

Through this work, we aim to answer several key questions related to bb-MINLP and propose a new algorithm that can solve bb/gb-MINLP problems of moderate sizes (i.e., up to 15 variables and 23 constraints). First, to solve bb/gb- MINLP problems, we propose the use of mixed-integer surrogate models that can handle discrete variables directly, rather than relaxing the integrality constraints. The framework utilizes, compares, and combines two types of surrogate models, namely Artificial Neural Network (ANN) and Gaussian Process (GP) models. A data-preprocessing technique, one-hot encoding, is used to address the modeling of mixed-variable problems, and the optimization problem is reformulated to reflect this transformation. In addition, we study the performance of three different sampling strategies for mixed-integer problems and propose the most appropriate method that balances solution accuracy and sampling requirements. Finally, we develop an algorithm that can solve both black- and gray-box formulations of (P1) using a hybrid combination of ANN and GP models for the MINLP and NLP stages, respectively. The performance of the algorithm is analyzed with respect to solution accuracy, sampling requirements, and computational efficiency and compared to those of two competing existing algorithms for bb-MINLP.

This chapter is organized as follows. Section 2 introduces the necessary background on surrogate-based optimization and reviews some existing work on bb-MINLP. In Section 3, sampling, data-preprocessing, and surrogate modeling strategies for bb-MINLP are presented in detail. The overall proposed algorithm is described in Section 4 to illustrate

how these strategies are integrated into the overall framework. Section 5 presents a comprehensive comparison of the proposed methodology on a set of benchmark problems. Finally, Section 6 introduces the surrogate formulation for a case study on superstructure optimization. We demonstrate how the MINLP problem can be decoupled into a gray-box problem and report the performance of our algorithm. A discussion of the findings is provided before the conclusions and future perspectives. The detailed formulations of the process synthesis case study are provided in the Appendix.

4.2 Overview of Surrogate-Based Optimization

4.2.1 Existing Literature on Derivative-Free MINLP Optimization

Existing work on derivative-free MINLP optimization algorithms can be divided into three broad categories: sampling-based/direct-search, model-based, and stochastic or evolutionary methods [11]. Direct-search bb-MINLP algorithms have been proposed as extensions to existing NLP direct-search methods[52-54,109,116]. One of the most popular existing software for constrained bb-MINLP is NOMAD (Nonsmooth Optimization by Mesh Adaptive Direct Search) adapted from [49].

Due to their purely sampling-based nature, evolutionary-type methods (i.e., Genetic Algorithms[117], Particle Swarm Optimization[118], and Simulated-Annealing[119]) could also be applied to solve (P1). The most widely used algorithm for bb-MINLP problems under this category is developed in a MATLAB "global optimization" toolbox, which employs a genetic algorithm[64,110]. In addition, MEIGO[120] is an open source software tool that uses enhanced scatter search for global optimization of NLP and MINLP formulations.

There have also been a few developments for solving bb-MINLP problems in the surrogate-based optimization literature. Existing surrogate-based MINLP algorithms start with relaxing the discrete variables to create smooth surrogate functions. For example, SO-MI introduced in [55] and MI-SO[62] use a cubic radial basis function (RBF) model to solve expensive black-box problems. An RBF-based algorithm for mixed-integer nonlinear constrained optimization has been proposed in [59] and [60]. Both methods have been shown to perform well for problems up to 8 binary and 4 continuous variables and less than 10 constraints. All of these aforementioned surrogate-based MINLP optimization algorithms do not handle discrete variables directly. Instead, the integrality constraint is relaxed to construct a smooth surrogate model, even if the simulation cannot be inquired at non-integral locations of the discrete variables. Recently, the use of gradient-boosted tree has been proposed for mixed-integer convex nonlinear optimization [121]. While this method handles discrete variables directly, the resulting gradient-boosted tree model is discontinuous. Additional relevant work from the process systems engineering community involve optimization of MINLP formulations with embedded surrogate functions [15,106,107]. However, in these contributions, the discrete variables are decoupled from the surrogate models. All surrogate models are only a function of continuous variables, and the trained surrogate models are embedded in the overall MINLP formulation.

In the approximation literature, several efforts have been made to study mixed-variable surrogate models that can directly handle discrete variables [115,122,123]. These surrogate model techniques have been studied only with respect to approximation accuracy, and the optimization of these models has not been studied. This is very important because although certain functions might be accurate approximations, the formulation that

needs to be embedded within the surrogate-based optimization problem may lead to intractable problems. For example, in [122], a Gaussian process with a special correlation function is proposed. This correlation function models interactions between discrete-discrete and discrete-continuous variables. We have found that optimizing this model requires many additional constraints to allow the selection of appropriate correlation coefficients, in addition to the number of constraints that increases proportionally with the number of samples that are used to construct the model. Thus, this leads to a very large surrogate bb-MINLP that is very challenging to optimize even with state-of-the-art deterministic optimization solvers. Another way to construct mixed-integer surrogate models is using regression trees as proposed in [123]. Since regression tree methods involve dividing the search space into several partitions, this allows for a natural development of different regression functions for different realizations of discrete variables. For each partition, or a "node" of a tree, a Gaussian process model can be constructed. While this method is straightforward and easy to adapt, the resulting optimization problem is a piecewise function, which may require a generalized disjunctive programming formulation for its optimization. As these models were developed solely for the purpose of prediction, the optimization of these mixed-integer surrogate models may be infeasible or difficult. In this work, we have limited our study to methods that balance accuracy and tractability of formulation for optimization.

4.2.2 Methods for Surrogate-based bb-MINLP

4.2.2.1 Design of mixed-variable computer experiments

The accuracy of a surrogate model depends both on the number and the location of points. Hence, finding a good initial sampling design is an important step that highly affects the accuracy of the final solution [5,7,124]. When only continuous variables are present, LHS is typically used to generate an initial sampling design[1,2,20]. However, for MINLPs, the standard way of obtaining a space-filling sampling design is an open question that we aim to study in this work.

To illustrate some existing sampling methodologies, let us define n_{lhs} to be the number of points selected for each LHS and m to be the total number of discrete combinations. If all variables are continuous, the total number of LHS points is typically fixed to a number based on heuristics (e.g., $n_{lhs} = 10k_1 + 1$). If L_j represents the number of discrete levels for each discrete variable y_j (e.g., a binary variable y_j has two levels: $L_j = 2$), then $m = \prod_{j=1}^{k_2} L_j$. In [55,62], the LHS points corresponding to discrete variables are rounded to closest integers. In [59] and [60], an auxiliary problem is solved to eliminate infeasible samples. These approaches, however, do not guarantee that the same number of samples is collected for each discrete realization. Thus, this may lead to data imbalance: certain discrete combinations may contain more data points than the others, which may lead to inconsistency in the accuracy of the surrogate model for certain levels.

Swiler et al. [115] propose and compare different sampling techniques, such as standard Latin hypercube and *k*-Latin hypercube sampling for building accurate approximation models. When using the standard Latin hypercube approach, one LHS of size mn_{lhs} is generated in the continuous space, and the points are then randomly split into m groups of equal size sets, so that each group is assigned to a unique discrete level. In the

k-Latin hypercube approach, a separate LHS of size n_{lhs} is generated for each discrete level. Both methods generate mn_{lhs} points. Although these methods have been compared for their approximation accuracy of low-dimensional functions, they have not been systematically compared with respect to their performance for surrogate-based optimization. In this work, we compare the performance of three different sampling strategies from the approximation and surrogate-based optimization literatures: k-Latin hypercube (Sampling Strategy 1), standard Latin hypercube (Sampling Strategy 2), and Latin hypercube sampling with simply rounding any discrete variables to their nearest integer value (Sampling strategy 3).

4.2.3 Mixed-integer Surrogate Model Construction via One-hot Encoding

In this chapter, we use both ANN and GP for surrogate models. The simplest way to build surrogate models for mixed-variable functions is to assume that all k_2 discrete variables are continuous and proceed with training an ANN or a GP model with k inputs. This approach has been used in the model-based surrogate optimization literature thus far [55,59,60,62]. This approach is attractive due to its simplicity, but it assumes that all inputs are continuous and ordinal. Instead, we propose an alternative way of constructing a mixed-variable surrogate model without relaxing the integrality constraint through the use of one-hot encoding [125]. One-hot encoding involves converting binary or integer variables to dummy variables. Through one-hot encoding, we can make sure that regardless of the values of the discrete variables, the effect of this input on the output prediction does not diminish [126]. For example, Figure 14 shows a simple ANN with one hidden layer with 3 nodes and 1 output node. The original problem has one continuous (x) variable and one binary (y) variable. If we are using a standard ANN, the functional form will be:

$$h_l(\boldsymbol{x}) = \sigma\left(w_{x,l}x + w_{y,l}y + b_l\right), l = 1,2,3 \tag{13}$$

$$\hat{f}_{NN}(\boldsymbol{x}) = \sigma\left(w_{1,f}h_1 + w_{2,f}h_2 + w_{3,f}h_3 + b_{out}\right) \tag{14}$$

where h_l represents the three nodes in the hidden layer, $w_{s,d}$ represents the weight of the ANN (s = source and d = destination), and b_l and b_{out} are bias values of the hidden and the output layers, respectively. If $y = 0$, then all signals from the binary variable node y will become zero and all of the terms $w_{y,l}y$ in Eq. 4 will be equal to zero. On the other hand, if $y = 1$, then all signals from node y will now become one. This is problematic as $\{0,1\}$ does not represent an intensity, and $y = 0$ might represent a certain effect on the output that must be captured (e.g., the presence of a process unit or not).

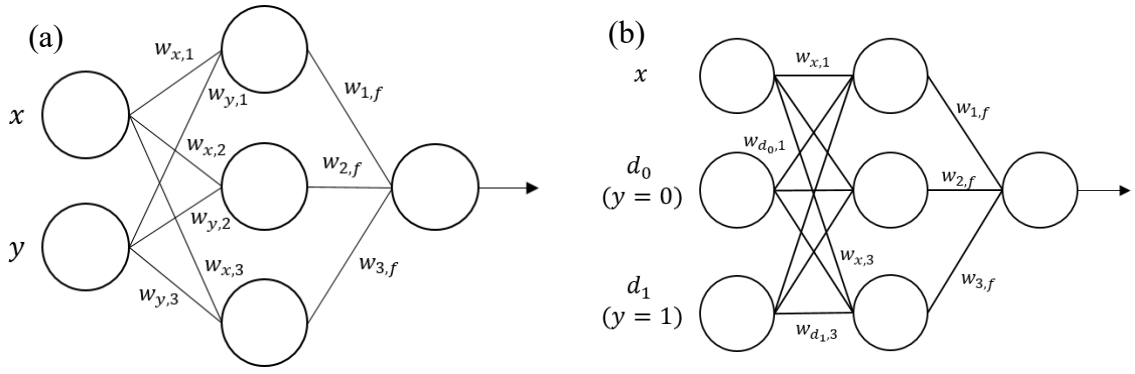


Figure 14. One-hot encoding for a neural network with one continuous x and one binary y variable

One-hot encoding can be used to overcome this problem and improve model accuracy in the case of mixed-variable problems. One-hot encoding converts the original binary variable into two dummy variables, each representing a distinct value of the original binary variable. Consequently, the structural complexity of the model (i.e., number of input

nodes in the neural network) increases due to additional dummy variables. If we have one binary variable ($y = \{0,1\}$), two dummy variables are created as follows:

$$d_0 = \begin{cases} 1 \; if \; y = 0 \\ 0 \; if \; y = 1 \end{cases}, d_1 = \begin{cases} 0 \; if \; y = 0 \\ 1 \; if \; y = 1 \end{cases} \tag{15}$$

The resulting functional form of the ANN is:

$$h_l(\boldsymbol{x}) = \sigma\left(w_{x,l}x + w_{d_0,l}d_0 + w_{d_1,l}d_1 + b_l\right), l = 1,2,3 \tag{16}$$

$$\hat{f}_{NN}(\boldsymbol{x}) = \sigma\left(w_{1,f}h_1 + w_{2,f}h_2 + w_{3,f}h_3 + b_{out}\right) \tag{17}$$

Thus, depending on the value of y, only one of d_0 or d_1 is active. While the dummy variables are still discrete, one-hot encoding allows that the effect of variable y on of $h_l(x)$ is represented evenly regardless of the value of y by allowing either d_0 or d_1 to be always equal to one. As a result, the overall signal from the binary node remains undiminished regardless of the value of the binary variable. For the case of integer variables, the surrogate model could be constructed without one-hot encoding if integer variables represent ordinal relationships. The integer values can then be scaled between 0 and 1 before constructing a surrogate model [55]. If integer variables do not represent ordinal relationships, one-hot encoding can be used to represent the different integer variable levels, or the integer variables can be transformed to binary variables. Subsequently, Eq 6 can be used directly [127].

To optimize a model with one-hot encoding, we need to add an additional constraint to make sure only one of the dummy variables is selected (i.e., $d_0 + d_1 = 1$). Throughout

this chapter, the surrogate model generated using one-hot encoding will be noted as "mixed-integer (MI)" surrogate; the one generated without one-hot encoding (i.e., relaxing the integrality constraint) will be noted as "relaxed (RE)" surrogate. One-hot encoding is performed during the data-processing stage, where the dataset for binary variable is transformed into dummy variables. For example, if the original dataset is $[\boldsymbol{X}, \boldsymbol{Y}]$, where $\boldsymbol{X}$ represents a set of data for continuous inputs and $\boldsymbol{Y}$ represents a set of data for binary inputs, the transformed dataset is $[\boldsymbol{X}, \boldsymbol{D_0}, \boldsymbol{D_1}]$, where $\boldsymbol{D_0}$ and $\boldsymbol{D_1}$ represent each dummy inputs created for each binary value. After this transformation, a MI surrogate model is constructed using either ANN or GP using the transformed, augmented dataset.

4.3 Proposed Algorithm

In order to solve (P1), the MI and RE surrogate models are integrated into a black-/gray-box optimization framework described previously. The overall algorithm can be decomposed into two main steps: 1) MINLP search, and 2) NLP search. Surrogate models are constructed in both search steps for all black-box constraints, and both the MINLP and NLP search steps are illustrated in detail in Table 7 and Table 8. First, all black-box equality constraints h_{EB} are transformed into two inequalities and are added to set IB. The MINLP search is first performed to find a solution with respect to all variables (i.e., both continuous and discrete variables). The NLP search is then performed by fixing discrete variables at optimal values determined from the MINLP step and optimizing only with respect to continuous variables. The NLP search step allows the algorithm to further reduce constraint violations and refine the incumbent solution. The overall algorithmic steps for both the MINLP and NLP search stages consist of three main steps: 1) initial sampling, 2) surrogate modeling, and 3) optimization and adaptive sampling. The main differences between the

MINLP and NLP step are: (a) whether one-hot encoding is used to construct a mixed-integer surrogate model or not, and (b) how the incumbent solution is selected at each iteration. The general framework is written in Python and the optimization is performed via an in-house Python-GAMS interface.

4.3.1 Initial Sample Design

For simulation-dependent optimization problems, choosing an initial sampling design is important since the simulation may fail to converge if a continuous value is used instead of a discrete value. Three sampling strategies are compared in this work as described earlier. For all sampling strategies, we generate a total of $m \times max\left(5, \left\lceil \frac{10(k_1+k_2)+1}{m} \right\rceil\right)$ points. The minimum number of 5 points is a heuristic on the minimum number of points that must be included in each level to ensure that at least this many points are sampled from each level. All collected samples are scaled between 0 and 1 using x_i^l and x_i^u before proceeding to fit any surrogate models. The three sampling strategies (SS1, SS2, SS3) are illustrated in Table 6.

Table 6. Three sampling strategies for the initial design of experiment

Algorithm 1: Initial Design of Experiment
Input: problem dimension k, continuous dimension k_1, binary dimension k_2, total number of levels $m = \prod_{j=1}^{k_2} L_j$ **Output**: Latin hypercube design $\boldsymbol{S_{lhs}}$ **Initialization**: $n_{lhs} = \max\left(5, \left\lceil \frac{10k+1}{m} \right\rceil\right)$
Sampling Strategy 1 (SS1):
$\boldsymbol{S_{lhs}} \leftarrow [\,]$ for $i = 1\ to\ m$ do Generate a Latin hypercube design $\boldsymbol{X}$ of size $n_{lhs} \times k_1$ for $\boldsymbol{x}$ Construct a dataset $\boldsymbol{Y}$ of size $n_{lhs} \times k_2$, where all rows represent a single unique combination of binary variables $\boldsymbol{S_{lhs}} \leftarrow [\boldsymbol{X}, \boldsymbol{Y}]$ End
Sampling Strategy 2 (SS2):
Generate a Latin hypercube design $\boldsymbol{X}$ of size $mn_{lhs} \times k_1$ for $\boldsymbol{x}$ for $i = 1\ to\ m$ do Randomly select n_{lhs} rows from $\boldsymbol{X}$ Construct a dataset $\boldsymbol{Y}$ of size $n_{lhs} \times k_2$, where all rows represent a single unique combination of binary variables $\boldsymbol{S_{lhs}} \leftarrow [\boldsymbol{X}, \boldsymbol{Y}]$ End
Sampling Strategy 3 (SS3):
Generate a Latin hypercube design $\boldsymbol{S} = [\boldsymbol{X}, \boldsymbol{Y}]$ of size $mn_{lhs} \times k$ For k_2 columns, round the values to the closest integer: $\boldsymbol{S_{lhs}} \leftarrow [\boldsymbol{X}, \lceil \boldsymbol{Y} \rceil]$
return S

4.3.2 Surrogate Model Construction

During this stage, surrogate models are developed in the scaled domain using either an ANN or GP to represent each of the outputs (i.e., objective function and unknown constraints). For mixed-integer surrogates, one-hot encoding is performed to convert the original binary variables to dummy variables. 10-fold cross-validation is used to find the best model during each iteration of the overall algorithm. While 10-fold cross-validation

allows the algorithm to construct a model that generalizes well to a new set of data, one disadvantage of k-fold cross validation is the increased computational cost due to training k models at each iteration. In our work, we have observed that the CPU time for model construction is negligible relative to the optimization CPU time. Nevertheless, when model construction becomes computationally more expensive, surrogate fitting could happen in parallel using multiple processors to reduce the CPU time.

For the ANN, both the objective and all constraints are modeled simultaneously by using multiple output nodes (Multiple Input – Multiple Output ANN). In our work, we use a hyperbolic tangent function as an activation function; for the final layer, a linear activation function is used. As our goal is to locate a global optimum rather than finding a perfect surrogate representation, we are not necessarily interested in finding a good approximation in regions of low interest (i.e., areas far away from global optimum). Thus, the balance between model accuracy and sampling requirement is achieved by keeping the overall model complexity low while maintaining high accuracy in regions of high interest (i.e., areas where the global optimum is likely to be located). Instead of using an extensive search methodology to find the number of hidden layers and hidden nodes, we use a simple heuristic to determine the number of nodes in a hidden layer. Specifically, only one hidden layer is used and the number of nodes is 2/3 of the number of input nodes plus the number of output nodes (i.e., total number of constraints and the objective) [67]. This strategy allows us to locate a good optimum within a reasonable computation time. While methods such as grid search and stochastic optimization may lead to a more accurate ANN, it can be computationally too expensive for surrogate-modeling, especially when several iterations are required to converge to a solution. After the optimal hyperparameters are determined,

we then compute optimal weight and bias values for the network using back propagation. For GP models, the training procedure does not require the selection of hyperparameters, but just the optimization of the surrogate model parameters. Another distinction is that all constraints and the objective are fitted separately using a multiple-input single output approach.

4.3.3 Surrogate Model Optimization and Adaptive Sampling

After constructing surrogate models for both the constraints and objective, an optimization problem is formulated. For ANN, the hyperbolic tangent function needs to be reformulated since optimization solvers cannot handle hyperbolic tangent functions. As suggested in [21], the hyperbolic tangent function is reformulated as $tanh(x) = 1 - \frac{2}{e^{2x}+1}$ since it was shown to outperform other reformulations. When a mixed-integer surrogate model is used, an additional constraint is needed to allow the selection of only one dummy variable for each binary variable. This can be formulated into a simple linear constraint: $d_0 + d_1 = 1$. For a gray-box problem, where we can assume certain constraints are known, we need to formulate and scale the gray-box constraints accordingly (see Section 6). A diverse set of local and global solutions are collected using global and multistart local optimization using BARON [83] and DICOPT [128] solvers, respectively. This approach aims to find a balance between exploration and exploitation and avoids premature convergence to a local optimum.

In some instances, surrogate models might have failed to accurately approximate the constraints within the entire search space. As a result, the resulting surrogate optimization formulation is infeasible, even though this does not immediately imply that

the original problem is infeasible. When this occurs, the algorithm then solves an infeasibility problem to locate the most feasible solution with the least constraint violation. Instead of minimizing the surrogate objective ($\hat{f}(\boldsymbol{x}, \boldsymbol{y})$) subject to surrogate constraints $\left(\hat{g}(\boldsymbol{x}, \boldsymbol{y})\right)$, we minimize the sum of slack variables s_c, as shown in (18).

$$\begin{aligned} &\min \sum_{c=1}^{C} s_c \\ &s.t.\ \hat{g}_c(\boldsymbol{x}, \boldsymbol{y}) - s_c \leq 0, \quad c = 1, \dots, C \\ &0 \leq s_c \leq 0.1, c = 1, \dots, C \end{aligned} \tag{18}$$

In the above formulation, set c represents unknown inequality constraints and $\hat{g}_c$ represents the surrogate approximations of all unknown inequality constraints.

All local and global solutions are added to the sampling set and the best incumbent solution is found at each iteration by calculating a score. The solutions are ranked in ascending order based on the value of the objective function value (after sampling the simulation) $f(\boldsymbol{x}^*, \boldsymbol{y}^*)$ and the total constraint violation v. Consequently, among a set of all local and global solutions, the solution with the smallest objective function value gets the lowest objective function score ($\boldsymbol{S_{obj}}$); the solution with the smallest constraint violation gets the lowest constraint violation score ($\boldsymbol{S_{con}}$). Each solution is characterized by two scores, which is equal to their rank with respect to feasibility $\boldsymbol{S_{vio}}$ and objective function value $\boldsymbol{S_{obj}}$. The overall score is computed by averaging these two scores: $\boldsymbol{S} = \frac{(\boldsymbol{S_{con}} + \boldsymbol{S_{vio}})}{2}$. The solution with the lowest $\boldsymbol{S}$ score is chosen and added to the intermediate solution set.

By considering both $\boldsymbol{S_{obj}}$ and $\boldsymbol{S_{vio}}$, we hypothesize that we can achieve a balance between finding a global solution and finding a feasible solution when assessing the best solution found during each iteration.

These steps are repeated until one of the following termination criteria is met: 1) negligible constraint violation and model error (both $\leq 1e^{-5}$), 2) no improvement in the objective value over ten consecutive iterations, and 3) maximum number of samples is reached.

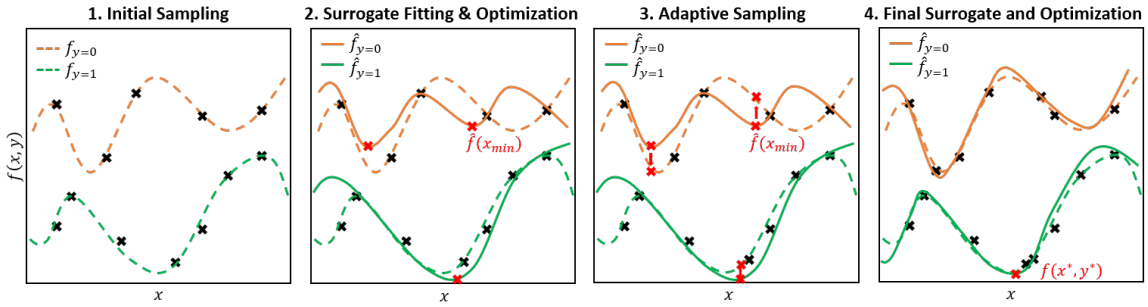


Figure 15. Illustration of adaptive sampling for a mixed-integer problem with one continuous x and one binary y variable. The true global optimum occurs when y=1. For each case of y, the true model exhibits a different behavior. At each iteration, all local and global solutions are collected and the simulation is re-inquired.

4.3.4 NLP Search

After the MINLP search step is complete, the algorithm proceeds to the NLP search to refine the best solution $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ found during the MINLP search step (Table 8). This step also allows us to further reduce constraint violations, a crucial step when many equality constraints are present. The discrete values are fixed at y^* and the solution is refined with only respect to the continuous variables. The overall algorithm for NLP step is similar to that of the MINLP search step, except that one-hot encoding is not performed and

CONOPT is used as a local solver [129]. In the final step, the algorithm reduces the bounds of all continuous variables to $\pm 1\%$ of the best solution found so far to further refine the solution. Assuming that the algorithm has already found an approximate solution, we only consider the constraint violation v to evaluate the solution quality during this final stage. The termination criterion of the NLP stage is identical to that of the MINLP stage.

Table 7. bb-MINLP optimization algorithm

Algorithm 2. Bb-MINLP optimization

Initialization: Initial Sampling

1. Create an initial LHS $\boldsymbol{S_{lhd}} = [\boldsymbol{X}, \boldsymbol{Y}]$ using the selected sampling strategy
2. Inquire simulation at $\boldsymbol{S_0} = [\boldsymbol{X}, \boldsymbol{Y}]$ to compute $\boldsymbol{Z_0} = f_{eval}(\boldsymbol{S_0})$. Assume one function evaluation provides the values of all constraints and the objective.

Data pre-processing

1. Scale $\boldsymbol{S_0}$ and $\boldsymbol{Z_0}$ between 0 and 1 and obtain $\boldsymbol{S_0'}$ and $\boldsymbol{Z_0}'$.
2. If fitting type = 'MI', perform one-hot encoding so that the binary variables y_j are transformed into dummy variables $d_{j,0}$ and $d_{j,1}$.

Surrogate model construction and optimization

Initialization: $\boldsymbol{S'} \leftarrow \boldsymbol{S_0'}, \boldsymbol{Z'} \leftarrow \boldsymbol{Z_0'}$

1. Use the chosen surrogate type to construct a surrogate model for all constraints and objective. Use 10-fold cross validation to find the best model.
2. Formulate the surrogate optimization problem and solve using both global and local solvers. If infeasible, solve infeasibility problem (P2).
3. Obtain p local and global solutions $\boldsymbol{S'_{new}} = [\boldsymbol{x_{new}}, \boldsymbol{y_{new}}]$.
4. Compute Euclidean distance between $\boldsymbol{S'_{new}}$ and existing sample set $\boldsymbol{S'}$:

$$dist_p = \frac{1}{k}\sqrt{\sum_{i=1}^{k_1}\left(X_{n,i} - x_{new,i}\right)^2 + \sum_{j=1}^{k_2}\left(Y_{n,j} - y_{new,j}\right)^2}$$

5. If $dist_p \geq 1e^{-10}$, unscale $\boldsymbol{S'_{new}}$ to the original bound and inquire simulation at $\boldsymbol{S_{new}}$ and compute $\boldsymbol{Z_{new}} = f_{eval}(\boldsymbol{S_{new}})$ and $\boldsymbol{v}$. Else, remove the solution from $\boldsymbol{S_{new}}$.
6. Compute the solution score for all collected intermediate solutions:
   a. $\boldsymbol{S_{con}} = rank(\boldsymbol{v}), \boldsymbol{S_{obj}} = rank(\boldsymbol{f_{new}})$
   b. $\boldsymbol{S} = \frac{(s_{con} + s_{obj})}{2}$
7. $f_{min} = argmin(\boldsymbol{S})$
8. If one of the convergence criteria is met, end iteration

Else, $\boldsymbol{S'} \leftarrow \boldsymbol{S_{new}}$ and $\boldsymbol{Z'} \leftarrow \boldsymbol{Z_{new}}$; repeat steps 1-2 for data preprocessing and steps 1-7 for surrogate model construction and optimization.

return $\boldsymbol{x}^*, \boldsymbol{y}^*$

Table 8. bb-NLP Algorithm

Algorithm 3: bb-NLP optimization

Input: Best solution found from MINLP search step $(\boldsymbol{x}^*, \boldsymbol{y}^*)$, variable bounds (x_i^l, x_i^u)

Initialization: LHS sampling

1. Check if previously sampled points $\boldsymbol{S'}$ can be reused. If yes, add to the sampling set.
2. Create an initial LHS $\boldsymbol{S_{0'}} = [\boldsymbol{x'_{lhd}}, \boldsymbol{y}^*]$ of size $10k_1 + 1$ only for continuous variables.
3. Un-scale initial LHS: $x_{lhd,i} = x'_{lhd,i}\left(x_i^u - x_i^l\right) + x_i^l$
4. Inquire simulation at $\boldsymbol{S_0} = [\boldsymbol{x_{lhd}}, \boldsymbol{y_{lhd}}]$ to compute $\boldsymbol{Z_0} = f_{eval}(\boldsymbol{S_0})$. Assume one function evaluation provides the values of all constraints and objective.

Data-Preprocessing

1. Scale $\boldsymbol{S_0}$ and $\boldsymbol{Z_0}$ between 0 and 1 and obtain $\boldsymbol{S'_0}$ and $\boldsymbol{Z_0}'$.

Surrogate model construction and optimization

Initialization: $\boldsymbol{S'} \leftarrow \boldsymbol{S'_0}, \boldsymbol{Z'} \leftarrow \boldsymbol{Z'_0}$

1. Use the chosen surrogate type to construct a surrogate model for all constraints and objective. Use 10-fold cross validation to find the best model.
2. Formulate the surrogate optimization problem and solve using both global and local solvers. If infeasible, solve infeasibility problem (P2).
3. Obtain p local and global solutions $\boldsymbol{S'_{new}} = [\boldsymbol{x_{new}}, \boldsymbol{y}^*]$.
4. Compute Euclidean distance between $\boldsymbol{S'_{new}}$ and existing sample set $\boldsymbol{S'}$:

$$dist_p = \frac{1}{k}\sqrt{\sum_{i=1}^{k_1}\left(X_{n,i} - x_{new,i}\right)^2 + \sum_{j=1}^{k_2}\left(Y_{n,j} - y_{new,j}\right)^2}$$

5. If $dist_p \geq 1e^{-10}$, unscale $\boldsymbol{S'_{new}}$ to original bound and inquire simulation at $\boldsymbol{S_{new}}$ and compute $\boldsymbol{Z_{new}} = f_{eval}(\boldsymbol{S_{new}})$ and $\boldsymbol{v}$. Else, remove the solution from $\boldsymbol{S_{new}}$.
6. Compute the solution score for all collected intermediate solutions:

$\boldsymbol{S_p} = rank(\boldsymbol{v})$

7. $f_{min} = argmin(\boldsymbol{S_p})$
8. If one of the convergence criteria is met, end iteration

Else, $\boldsymbol{S'} \leftarrow \boldsymbol{S_{new}}$ and $\boldsymbol{Z'} \leftarrow \boldsymbol{Z_{new}}$; repeat step 1 for data preprocessing and steps 1-7 for surrogate model construction and optimization.

return $\boldsymbol{x}^*, \boldsymbol{y}^*$

4.4 Results

The performance of the proposed bb-MINLP algorithm is first tested on a set of benchmark problems obtained from MINLPLib [130]. In order to evaluate the performance of the algorithm on the most difficult possible scenario (i.e., unknown objective function, equality and inequality constraints), we treat all problems as purely black-box systems. Table 9 shows the characteristics of the MINLP problems that are used as benchmarks in this work. Columns k_1 and k_2 represent the number of continuous and binary variables, and n_{Ieq} and n_{eq} are the number of inequality and equality constraints, respectively. The dimensionality of the problems ranges from 1-12 continuous variables ($k_1 \leq 12$) and 1-8 binary variables ($k_2 \leq 8$). The problems also have a varying amount of equality and inequality constraints, with the most challenging problem having 9 inequality constraints and 6 equality constraints.

Through the results shown in this work, we aim to study the performance of the proposed algorithm with respect to (a) the selection of a surrogate model type (i.e., ANN versus GP), (b) the use of one-hot encoding versus relaxation of integrality, and (c) the sampling strategy. A selection of the surrogate modeling type must be made for both the MINLP search stage and the NLP search stage of the algorithm. To compare different surrogate models, we have performed an analysis between a purely ANN-based and a purely GP-based versions of the proposed algorithm. However, based on findings described in the next section, we have also proposed a hybrid approach (i.e., the use of ANN models for MINLP search and GP models for NLP search). Thus, there are overall six approaches that are tested in this work with respect to the selection of the surrogate model as well as

the use of one-hot encoding (Table 10). Finally, we compare the performance of the proposed approach with two other existing algorithms for bb-MINLP. Specifically, a Mesh-Adaptive Direct Search algorithm (NOMAD [49]) and a Genetic Algorithm [110] are compared at their default settings.

Table 9. Names and descriptions of the MINLP test problems from MINLPLib [130]. The equality constraints are transformed into two inequalities.

Problem	k_1	k_2	n_{1eq}	n_{eq}
alan	4	4	5	2
ex1221	2	3	3	2
ex1222	2	1	3	0
ex1223	7	4	9	4
ex1223a	3	4	9	0
ex1224	3	8	5	2
ex1225	2	6	8	2
ex1226	2	3	4	1
fuel	12	3	9	6
gbd	1	3	4	0
gkocis	8	3	3	5
oaer	6	3	4	3
procsel	7	3	3	4
st_e13	1	1	2	0
st_e14	7	4	9	4
st_e15	2	3	3	2
st_e27	2	2	6	0
st_e29	3	8	5	2
synthes1	3	3	6	0

Table 10. Description of 6 surrogate model settings that are tested in Results. MI represents a mixed-integer surrogate model *with* one-hot encoding used in the MINLP stage, and RE represents a surrogate model with all variables assumed to be continuous.

Name	MINLP surrogate	NLP surrogate
ANNMI	ANN (w. one-hot encoding)	ANN
ANNRE	ANN	ANN
hyMI	ANN (w. one-hot encoding)	GP
hyRE	ANN	GP
GPMI	GP (w. one-hot encoding)	GP
GPRE	GP	GP

The algorithm is tested 3 times for each setting, and the best, average, and standard deviation of the solutions obtained are reported. The maximum number of samples allowed for each problem is 4000, since sampling is expensive in most simulation-based optimization studies and thus must be limited. Comparing both the best and the average results allows us to evaluate the performance of the algorithm and its consistency in locating global optimum. A problem is assumed to be solved if the relative error between the actual and predicted optimum is less than an error tolerance ($\epsilon = 0.01$), where: $\varepsilon_{obj} = \left|\frac{f_{actual} - f^*}{f_{actual}}\right| \leq \epsilon$. A very small constraint violation is allowed (i.e., $v \leq 1e^{-5}$), and this is mainly due to the difficulty in exact satisfaction of equality constraints in a black-box setting.

4.4.1 Selection of Surrogate Modeling Type

First, the performance of mixed-integer and relaxed surrogate models is compared for both ANN and GP (ANNMI, ANNRE, GPMI, and GPRE). In order to eliminate the effect of the sampling strategy for these experiments, we only use the *k*-LHS sampling approach (Sampling Strategy 1). In the proposed algorithm, the MINLP search step aims mainly to locate the optimal discrete solution, while the NLP search step aims to refine the solution with only respect to the continuous variables. Consequently, the performance of the MINLP search step is important, since the algorithm will converge to a local solution during the NLP step if an incorrect binary solution is found during the MINLP step. Figure 16 shows the percentage of correct binary solutions found by each method during the MINLP search step. For both ANN and GP surrogate models, the mixed-integer approach (i.e., the use of one-hot-encoding) allows the algorithm to more accurately locate binary

solutions. This is due to the increased prediction accuracy of the MI model. For all of benchmark problems, the discrete variables are binary variables, and these are not ordinal. The results indicate that one-hot encoding enables the algorithm to search more efficiently through the use of more accurate surrogate models that capture the effect of both the "0" and the "1" cases. Hence, even if the resulting optimization model is more complex (i.e., increased problem dimension due to additional dummy variables and additional constraints relating new variables), the MI approach still outperforms the relaxation approach for the sizes of problems we are solving in this work. If the number of binary variables further increases, then tractability issues may arise, a potential limitation of our proposed methodology.

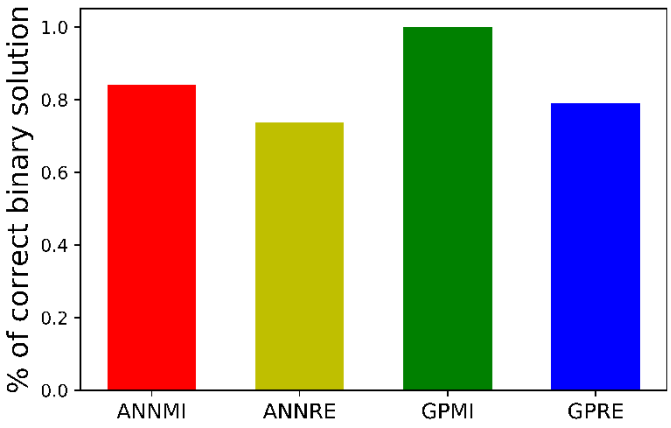


Figure 16. Performance of algorithm with respect to the capability to accurately locate correct discrete solution during the MINLP search step.

After the MINLP step terminates, the proposed algorithm proceeds to the NLP step. The goal of NLP step is to refine the solution obtained during the MINLP stage and locate the optimum while further reducing existing constraint violations. Figure 17 shows the performance profiles of the obtained best result (out of 3 repetitions) with respect to the number of samples and CPU time. The results show that GPMI outperforms the three other

methods, followed by GPRE. However, both GPMI and GPRE require more CPU time to converge compared to ANNMI and ANNRE. In addition, while ANNMI is able to better locate the correct discrete solution during the MINLP search step than ANNRE, Figure 17 shows that ANNMI and ANNRE solve the same number of problems. This implies that even when a correct discrete solution has been identified, the algorithm can still fail to locate a globally optimal solution during the NLP search step. This indicates that both the MINLP and NLP search steps are crucial in locating a global solution.

In addition to the best obtained result, it is important to assess the consistency of the methods. In Figure 18, the average and standard deviation of the three repetitions are reported with respect to solution accuracy and computational efficiency. When both the best and the average results are considered, it becomes clearer that the MI approaches on average perform better than RE approaches. This is attributed to the fact that the MI approaches locate the global binary solutions more consistently. For both ANNMI and GPMI, the average objective errors are smaller than those of ANNRE and GPRE, while the difference is much clearer for the ANN case. However, it is notable that while ANNMI has a smaller average objective error than that of GPMI (Figure 18), GPMI overall solves more problems (Figure 17). As we used strict criteria to generate a performance profile in Figure 17 (ϵ = 0.01 and $v = 1e^{-5}$), this result indicates that GP is better at accurately locating the solution and reducing constraint violations due to its interpolating nature. On the other hand, ANN models are good at finding the approximate location of the solution faster, but ANN models do not manage to continue improving the obtained solution and may converge to a local or infeasible solution. In addition, note that approximately 30% of the problems are not solved by any of the proposed methodologies. We observed that most

of these unsolved problems have several equality constraints. Thus, the exact satisfaction of constraint violation criterion ($v = 1e^{-5}$) becomes increasingly challenging. Nevertheless, we can test the limit of our algorithm by including these challenging problems in our benchmark problem set.

One notable difference between the performance of ANN and GP is the CPU time for model construction and optimization. The most time-consuming step when using GP is optimization, but its model construction CPU time is negligible. In contrast to GP, the majority of the computation time for ANN is attributed to model construction (i.e., optimization of parameters). Therefore, GP and ANN exhibit a very contrasting trend: GP models are easy to construct but difficult to optimize, while ANN models are time-consuming to construct but require less cost to optimize. This difference is due to their algebraic expressions: for ANN, the model expression depends only on the number of hidden nodes and layers, while for GP, the model expression is dependent on the number of data points used to construct the model. As a result, as the algorithm collects more samples during each iteration, the complexity of a GP model increases, and the CPU time for globally optimizing the model increases significantly, especially for the MINLP search step. Our results have confirmed that the majority of optimization CPU for the GP cases is resulting from the global optimization of the MINLP model. One potential suggestion to circumvent this problem is through the sole use of multi-start local optimization of the surrogate models. We performed this test and observed that the performance of the algorithm deteriorated; thus, we have concluded that global optimization of the surrogate models is important, because it leads to the location of new and more informative sampling solutions (i.e., better exploitation of the search space).

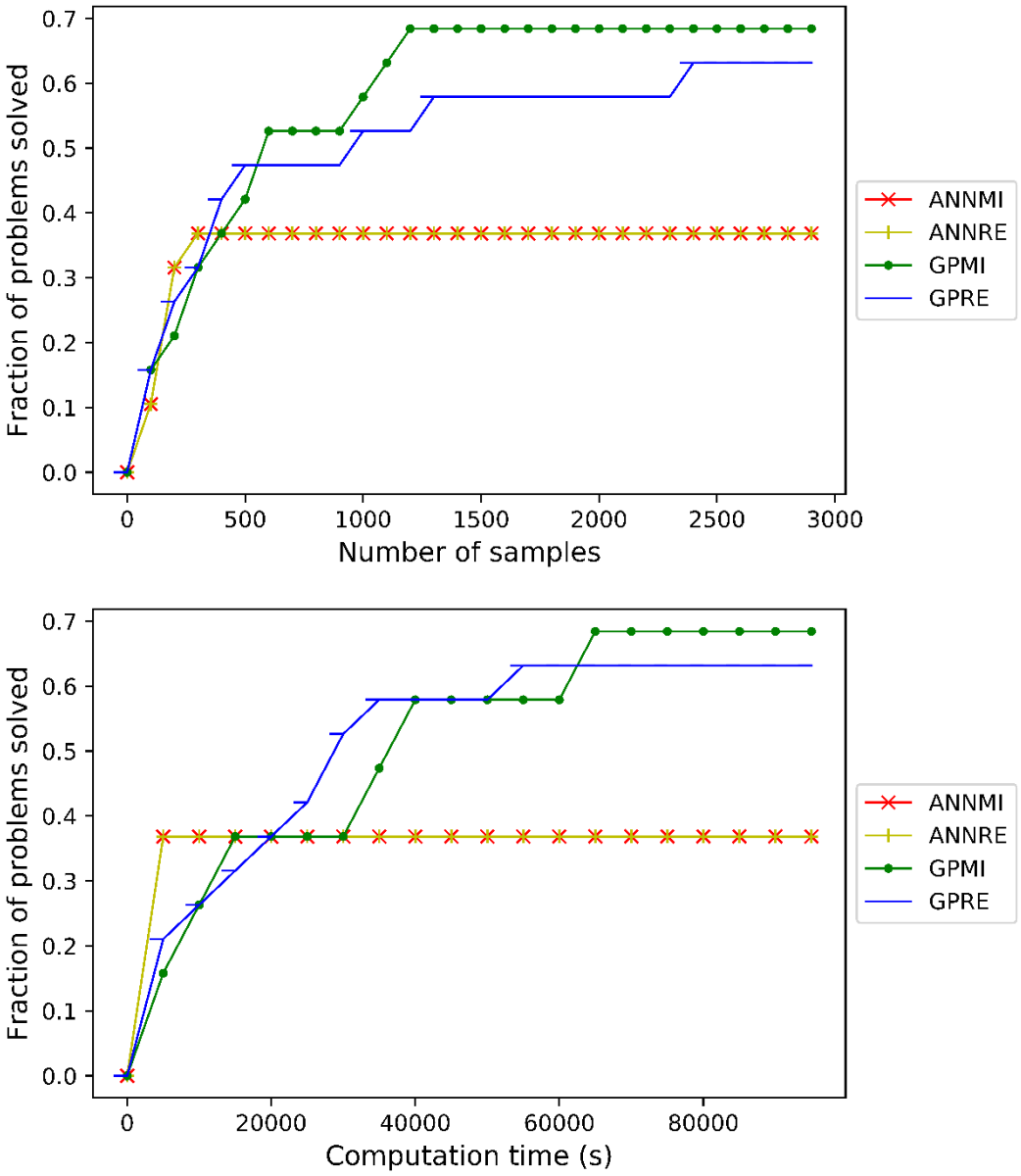


Figure 17. Performance profile of the best run for $\epsilon = 0.01$ and $v = 1e^{-5}$. The model types are described in Table 4.

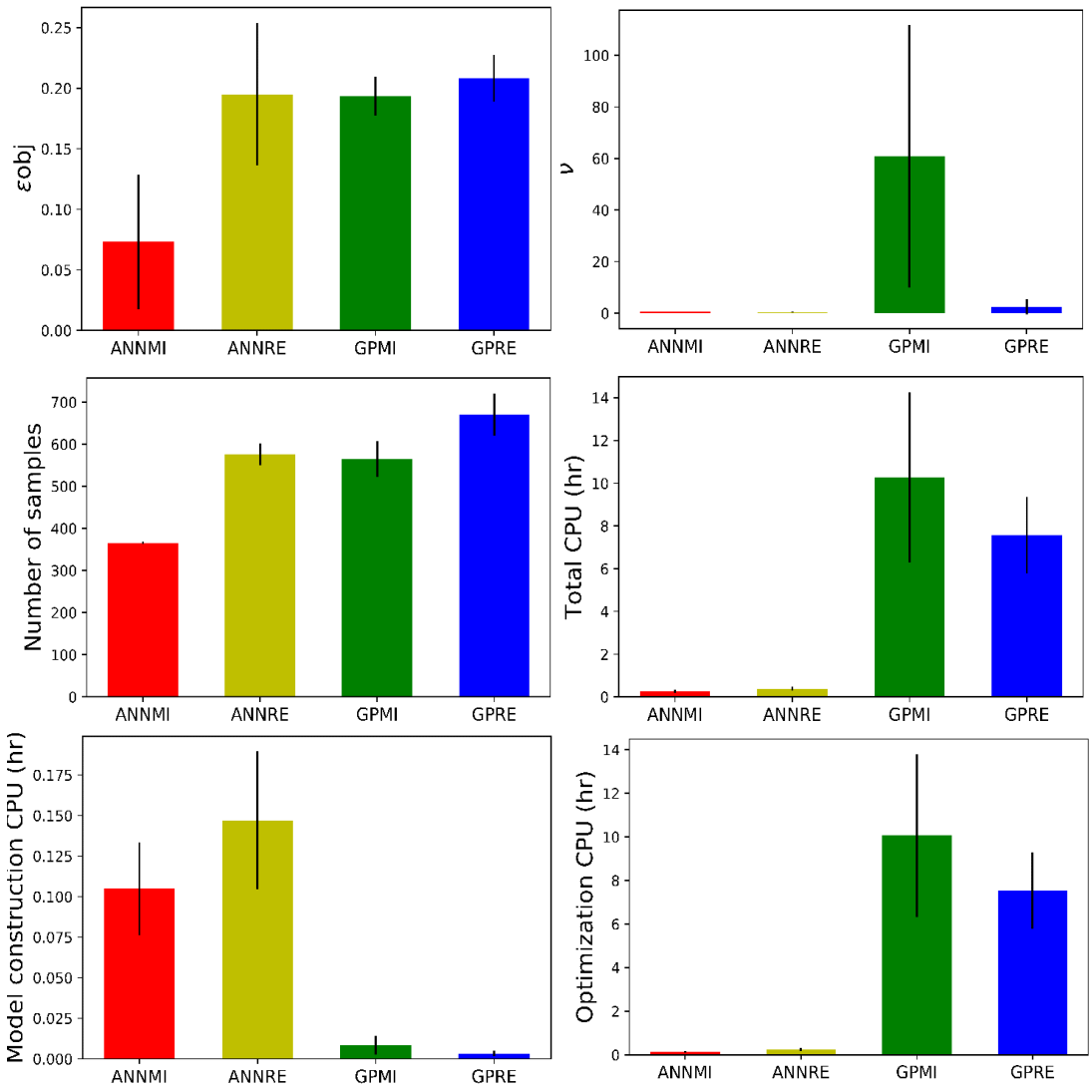


Figure 18. Statistics of three runs for MI and RE models. The standard deviation is plotted with the average to show the consistency of each method.

4.4.2 Hybrid Surrogate Modeling Approach

Based on the results obtained in the previous section, we have observed that ANN models have certain advantages over GP models due to their simpler functional form, which results in faster model optimization. On the other hand, all formulations with embedded GP models lead to more accurate approximations but are very difficult to

optimize. These observations led to the idea of exploiting the advantages of ANN and GP by creating a hybrid model, which combines ANN and GP. In particular, the algorithm uses ANNs for the MINLP search step, followed by GP for the NLP search step. Using ANN models initially allows the algorithm to expedite the MINLP search step and obtain a somewhat accurate solution, especially with respect to the discrete variables. Subsequently, since the aim of the NLP search is refinement, it is important to improve the accuracy of the final solution, which can be accomplished better using GP models. During the NLP stage, we have fixed the values of the discrete variables, so the dimensionality has been reduced and the algorithm suffers less from the computational expense of GP optimization. It is important to note that our algorithm has the capability to keep multiple potential discrete solutions, which will further be refined through the NLP search. However, in this work we have only shown its performance for the case where the best single solution is kept at the end of the MINLP search step. If a mixed-integer model with one-hot encoding is used for the MINLP step, the model is referred as "hyMI"; if not, we refer the model as "hyRE". In this work, we have not looked into the reverse case, where GP is used for MINLP step and ANN is used for NLP step, because this approach is expected to deteriorate solution accuracy and/or make optimization computationally more demanding.

We present the hybrid results (hyMI and hyRE) compared to those of ANNMI and GPMI, since the MI surrogate models were previously shown to perform better than the relaxed models. Figure 19 shows the performance profiles of the hybrid approach compared to ANNMI, GPMI, and existing solvers (GA and NOMAD). The same criteria are used to generate the performance profile: $\epsilon = 0.01$ and $v = 1e^{-5}$. As these methods heavily rely on sampling, they tend to require a large number of samples to converge.

However, GA and NOMAD algorithms require less CPU time than surrogate-based algorithms, because GA and NOMAD do not require the construction and optimization of surrogate models. However, when sampling becomes computationally expensive, GA and NOMAD are likely to be more time-consuming than surrogate-based approaches, since GA and NOMAD tend to require many function evaluations. It must also be mentioned that both NOMAD and GA contain a set of parameters that can be tuned to affect the performance of the algorithm, while in this work we compared all algorithms with their default settings.

When only the hybrid models are compared, hyMI outperforms hyRE. This is due to the capability of MI models to accurately locate binary solutions, which allows the algorithm to find a globally optimal as well as a feasible solution. After the correct binary solution is determined, the algorithm uses GP models to further refine the solution. The interpolating characteristic of GP is advantageous, particularly when the problem has several black-box equality constraints, because a good approximation of an equality constraint is crucial to find a feasible solution. Since the RE approach cannot find the correct binary solution as consistently as the MI approach, the hyRE model often converges to a local solution. As a result, the hyMI version of our algorithm combines all of the desirable characteristics and solves the most problems (about 80%), followed by GPMI and hyRE.

Figure 20 shows the average results of three runs with their associated standard deviations to illustrate the performance as well as the consistency of the algorithms. The hyMI approach overall performs the best with the smallest average objective error and constraint violation. It is also one of the most consistent methods suggested by its small

standard deviation. Compared to ANNMI, hyMI exhibits a significant improvement in solution accuracy. When the computation time is analyzed, hyMI can achieve a good balance between model construction and optimization CPU. Consequently, the optimization CPU for hyMI is significantly less than that of GPMI, which allows the algorithm to go through more iterations within a given time limit and further enhance the solution. The hyRE model performs better than ANNMI; however, it is outperformed by GPMI and hyMI since MI allows the algorithm to more accurately locate binary solutions. These results overall indicate that the MI approach using one-hot encoding outperforms the relaxation approach. Note that the hybrid approach has been proposed as a practical way of efficiently locating the solution within a reasonable CPU time. When no limit on computation resources exists, using GP for both the MINLP and NLP search steps may further improve the performance of the algorithm.

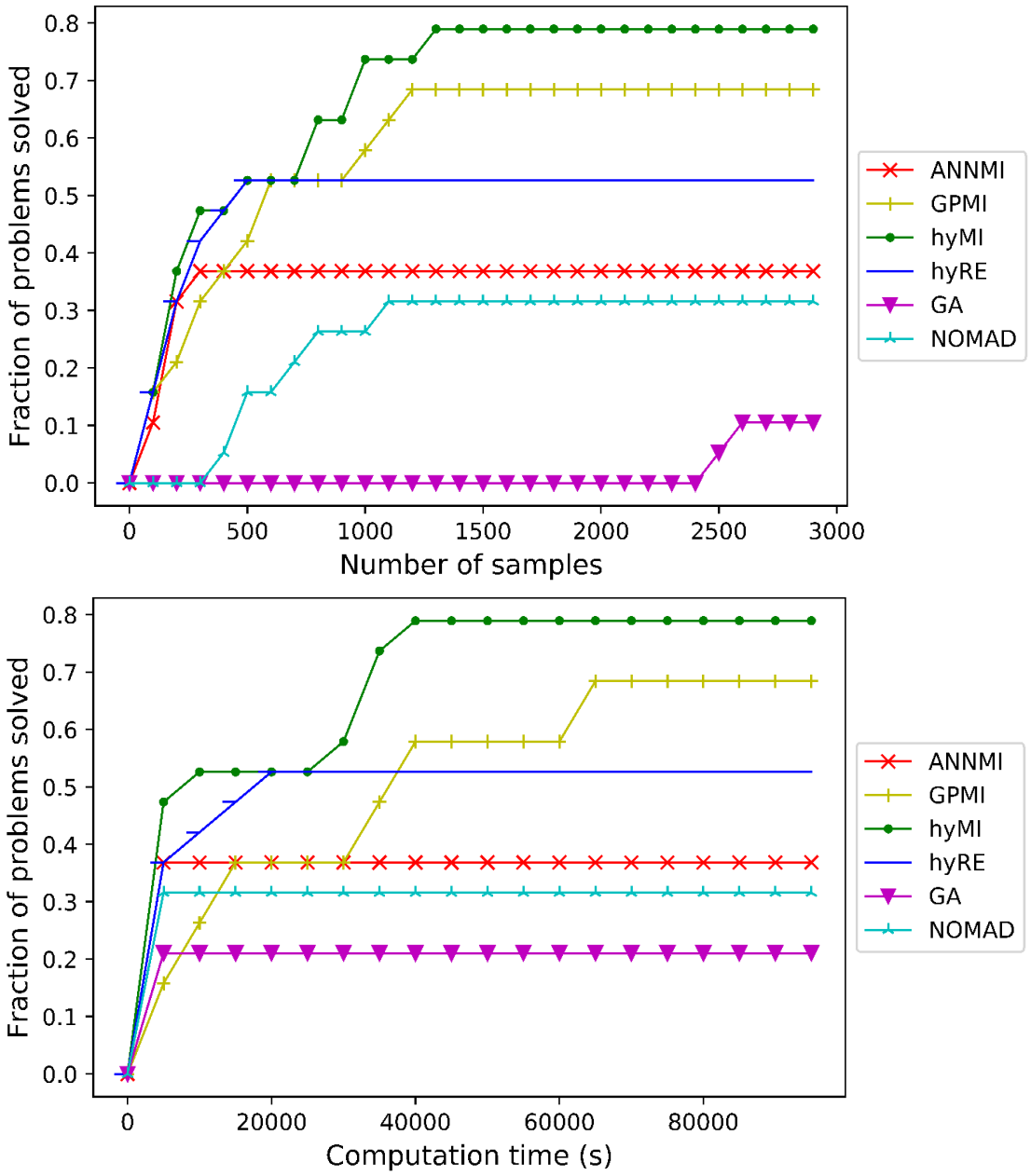


Figure 19. Performance profile of 4 surrogate types compared with existing bb-MINLP algorithms.

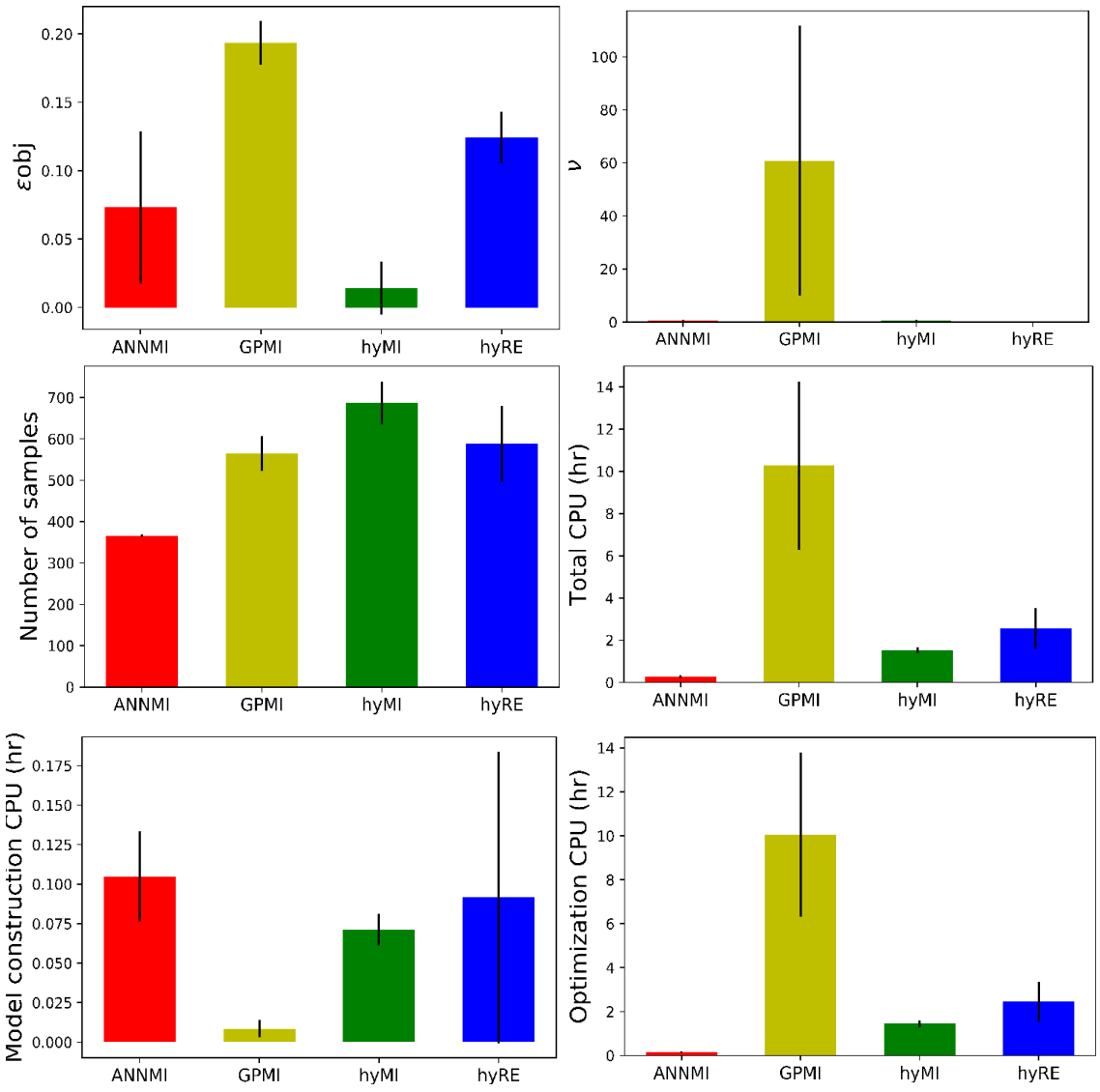


Figure 20. Average result of three runs for MI and RE models. The standard deviation is plotted with the average to show the consistency of each method. Hybrid MI approach shows the most consistent performance and solves the most problem.

4.4.3 Sampling Strategies for MINLP

Three sampling strategies are compared in this section while fixing all other algorithmic parameters: 1) *k*-Latin hypercube, 2) standard Latin hypercube, and 3) Latin hypercube sampling with simply rounding any discrete variables to their nearest integer value. As we previously have determined that hyMI outperforms all other methods, we use

the hyMI to test three sampling strategies (SS1, SS2, and SS3). Each sampling strategy is repeated 3 times and the best as well as the average results are calculated to evaluate its performance. Figure 21 shows a performance curve of the best result with respect to both the number of samples and the total computation time for $\epsilon \leq 0.01$ and $v \leq 1e^{-5}$. When only the best result is considered out of all runs, SS1 solves about 80% of the problems and outperforms other sampling strategies.

Figure 22 shows the average result of three runs with respect to solution accuracy (ε and v), the number of samples, and the computation time. Both the average and the best result indicate that SS1 outperforms all other sampling methods. Unlike SS2 and SS3, SS1 covers the entire search space evenly by generating a LHS of size n_{lhs} for all discrete levels. For SS2 and SS3, the points are randomly distributed into each discrete level, and thus the entire search space might not be represented evenly. Thus, even though all sampling strategies use the same number of samples, the ability to cover the entire search space evenly proves to be quite important for surrogate-based optimization. SS1 is also the most consistent approach, as shown by the small standard deviation values, while SS2 and SS3 are prone to data imbalance resulting from randomness.

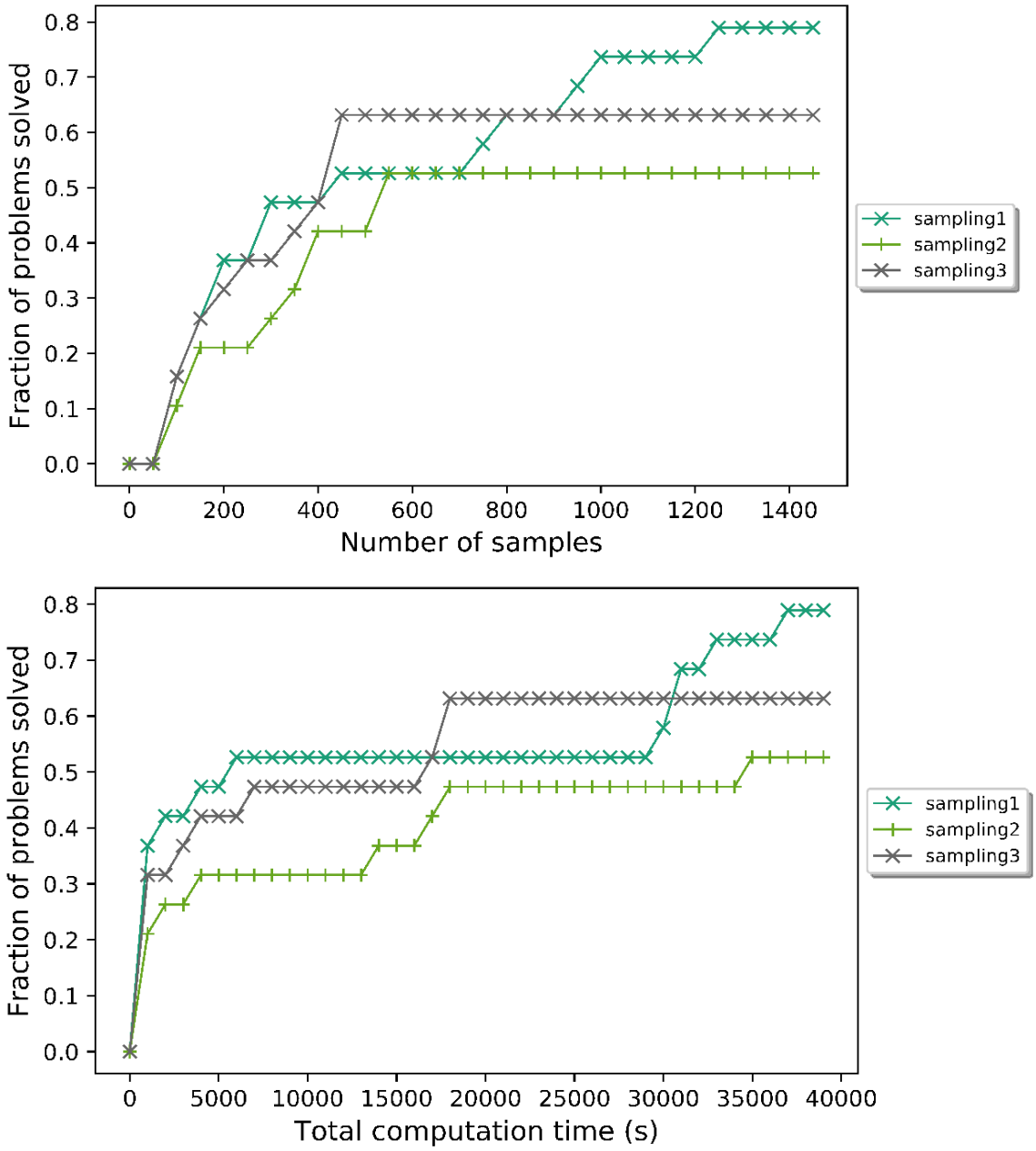


Figure 21. Performance curve of the best out of three runs for three sampling strategies for ϵ = 0.01 and v = 1e-5.

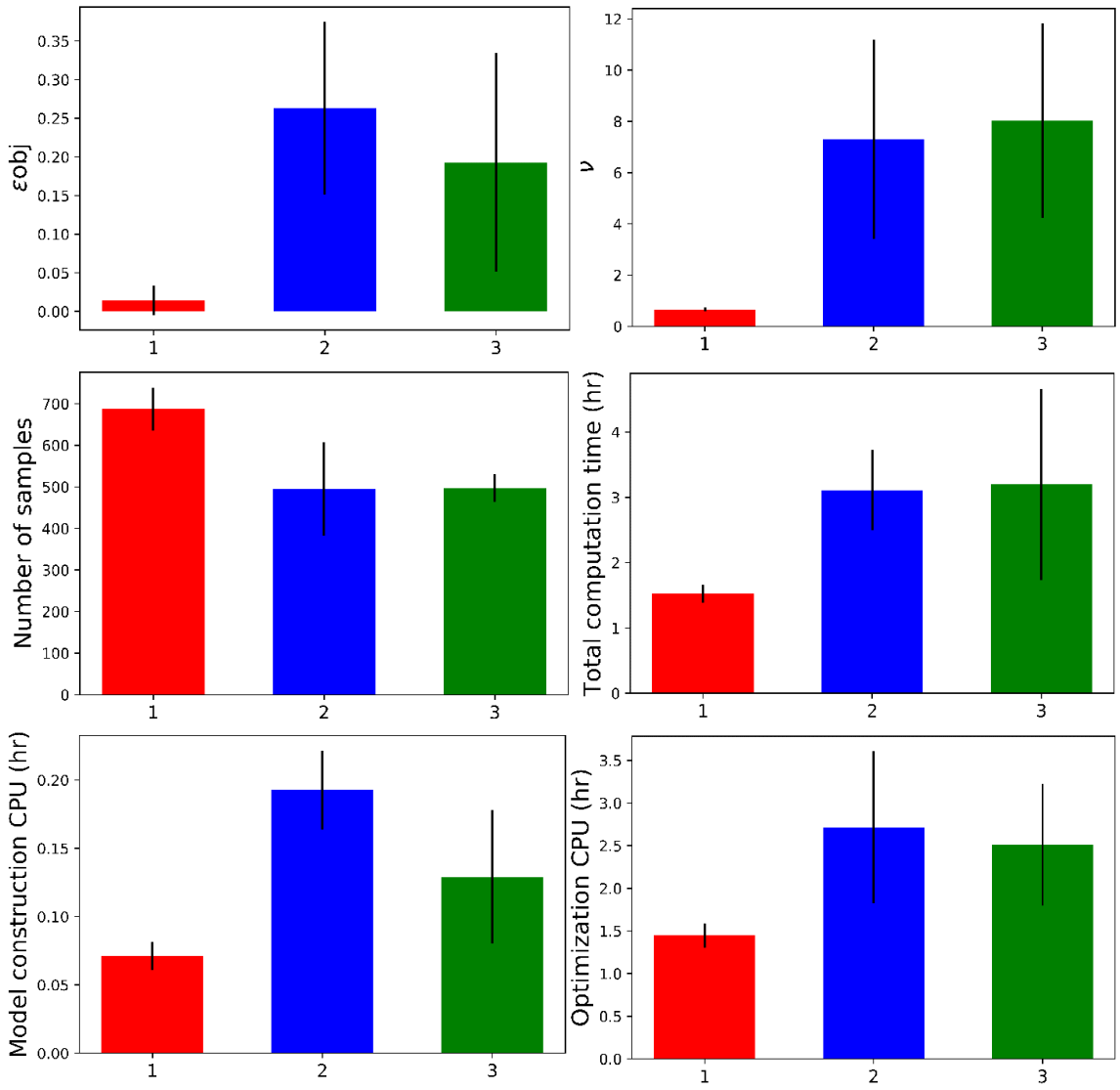


Figure 22. The average result of three sampling strategies (SS1=1, SS2=2, SS3=3) with respect to solution accuracy and computational efficiency.

4.5 Gray-box MINLP: a Process Synthesis Case Study

Up to this point, we have tested our algorithm for a black-box case, assuming all constraints and objective are unknown. In this section, we present a more challenging process synthesis case study and show how it can be solved as a gray-box MINLP formulation. When a problem has many constraints or variables, decoupling a problem into

a gray-box could reduce the complexity and improve the solution accuracy. This is representative of a typical process synthesis problem because constraints that connect individual units, such as material balance and logical constraints to control process synthesis, will typically be known a-priori. As the surrogate models are constructed in a scaled space, these known constraints must be scaled properly. Furthermore, when a mixed-integer surrogate model with one-hot encoding is used, the resulting optimization formulation is now in terms of dummy variables, instead of original binary variables. We will illustrate how the original constraints can be modified accordingly.

4.5.1 Problem Description

A superstructure optimization problem presented in [131] is used to demonstrate the proposed methodology. The objective is to determine the optimal structure and operating parameters for a process to minimize the sum of operating and capital costs. The binary variables are associated with each process unit, and continuous variables represent total mass flowrate. The nonlinearities in the model are due to nonlinear input-output relationship of process units. Although this benchmark problem contains simplified nonlinear relationships to represent process units, in a real case study, these nonlinear relationships represent the underlying phenomena captured by expensive simulations. The original MINLP formulation has 9 continuous and 8 binary variables with 23 constraints, and the original problem is shown in Appendix A. The superstructure is shown in Figure 23.

For process synthesis problems, we can safely assume that certain constraints related to mass conservation and process configuration will be known explicitly. For

example, $y_1 + y_2 = 1$ enforces the selection of only one process unit; $1.25x_9 - 10y_3 \leq 0$ is a big-M constraint, which makes sure that a flow is set to zero when a process unit is not selected. These gray-box constraints can be handled explicitly (marked as (G) in Appendix A). Appendix B shows the re-formulation of the original problem, which is explained in the subsequent sections.

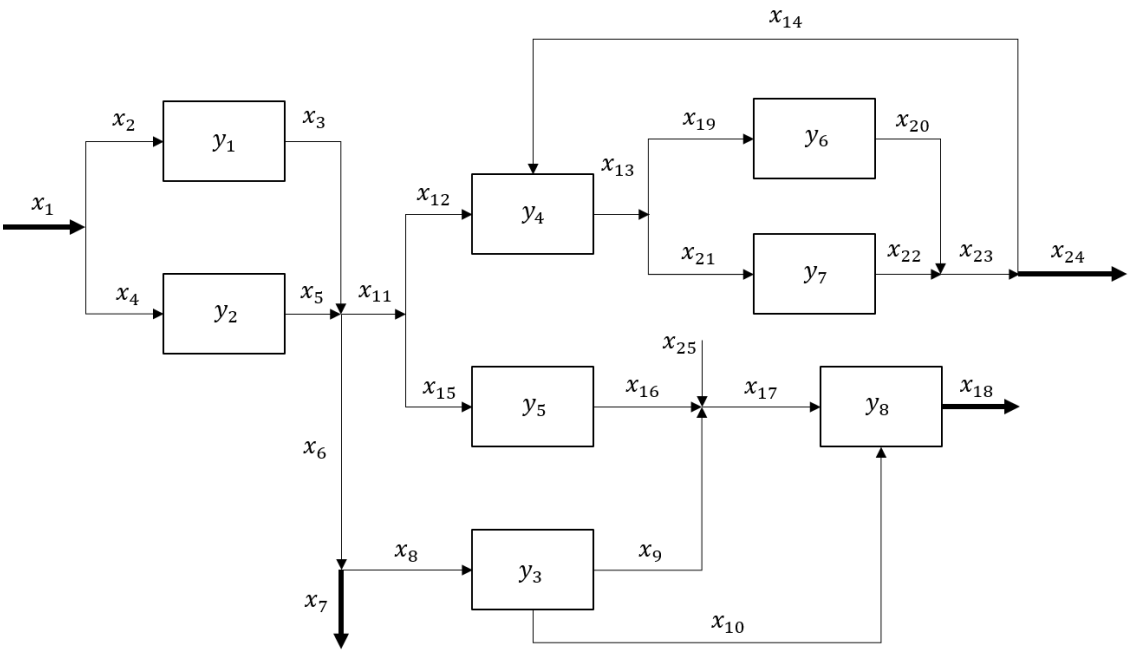


Figure 23. Superstructure of a process synthesis case study from [131] . Binary variables are used to represent 8 units, while continuous variables represent total mass flowrate.

4.5.2 Reformulation and Scaling of Gray-box Constraints

In order to generate an accurate surrogate model, normalization of input and output data is required. In this work, all input and output variables are scaled between 0 and 1 to construct a surrogate model. As a result, gray-box constraints need to be scaled properly so that optimization can now be performed in a scaled space. This is simple as all

continuous variables embedded within any known constraints can be scaled using the following transformation: $x_i = x_i'(x_i^u - x_i^l) + x_i^l$, where x_i is a continuous variable in the original domain, x_i' is a continuous variable in 0-1 scaled domain, and x_i^l and x_i^u are lower and upper bounds of the variable, respectively. No scaling is required for binary variables as they are already scaled between 0 and 1.

For MI surrogate models with one-hot encoding, the surrogate models are in terms of continuous and dummy variables, instead of original binary variables. As a result, gray-box constraints also need to be in terms of dummy variables. For a binary variable y_j with dummy variables $d_{j,0}$ and $d_{j,1}$ where $d_{j,0} = \begin{cases} 1 \; if \; y_j = 0 \\ 0 \; if \; y_j = 1 \end{cases}$ and $d_{j,1} = \begin{cases} 0 \; if \; y_j = 0 \\ 1 \; if \; y_j = 1 \end{cases}$, the following transformation is required for gray-box constraints:

$$d_{j,0}(1 - y_j) + d_{j,1} y_j = 1 \tag{19}$$

The original binary variable y_j can now be expressed as in terms of dummy variables:

$$y_j = \frac{1 - d_{j,0}}{d_{j,1} - d_{j,0}} \tag{20}$$

Furthermore, to allow the selection of only one dummy variable for each binary variable, an additional constraint is required for each y_j: $d_{j,0} + d_{j,1} = 1$. These are the overall set of steps that needs to be performed in order to re-formulate the original problem constraints, so that they are compatible with the surrogate-based formulation of our proposed algorithm.

4.5.3 Numerical Result

For the initial sampling design, SS1 is used. At each discrete level L_j, a Latin hypercube design of size n_{lhd} is generated. The input and output datasets are both scaled between 0 and 1, and hyMI model is constructed. As the problem consists of 9 continuous and 8 binary variables, the resulting neural network has 25 input nodes (i.e., 9 nodes for continuous variables and 16 nodes for dummy variables) and 18 output nodes for the objective and black-box constraints. In total, 6 out of 23 constraints are assumed to be known, while the rest are assumed to be unknown. This categorization of unknown and known constraints was performed such that only the very simple constraints (activation of flowrates and unit selection) are considered as known. A different decomposition of known and unknown constraints may lead to different results, with the expectation that more known constraints will help the algorithm converge faster. All gray-box constraints are scaled and reformulated with respect to dummy variables, and the final formulation is shown in Appendix B. Both global and local solvers are used to collect a diverse set of intermediate solutions. The model is optimized, and the algorithm terminates when one of the termination criteria is met.

Table 11 shows the final solution obtained using the proposed methodology, and Figure 24 shows the improvement in solution during each iteration. At the end of each iteration, we select the best solution based on both the objective function value and constraint violation v to find a solution that is not only globally optimal but also feasible. Note that at iteration 5, ε_{obj} temporarily increases because the algorithm found a more feasible solution with smaller constraint violation. During the next iteration, it quickly

converges back to the actual solution. After 19 iterations, the algorithm finds a global solution with less than 1% error and $v = 5e^{-4}$. After 22 iterations, the algorithm converges to an exact global solution with $\varepsilon_{obj} = 0$ and $v = 0$. The computational cost of this run is high and this is mainly attributed to both the training of the surrogate models and their optimization. However, the total number of samples required to solve this 17-variable problem is very low, considering the state-of-the-art in surrogate-based optimization. It should be noted that this problem is quite challenging when treated as a bb-MINLP problem. In fact, our proposed algorithm could not solve this problem within the given sampling and CPU limitations, when all constraints were treated as unknown. However, known constraints and the ability to incorporate them together with surrogate models, significantly facilitates the performance of the algorithm.

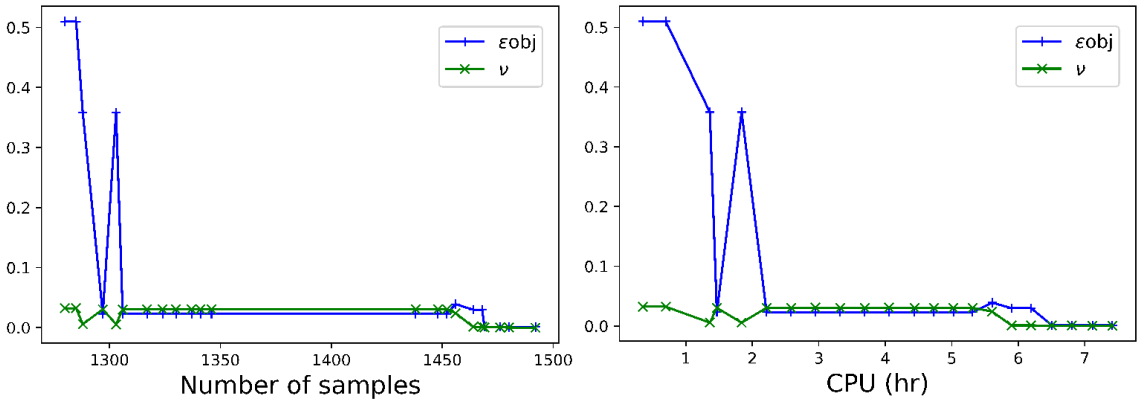


Figure 24. Solution error and constraint violation vs. the number of samples and computation time. Each point represents a single iteration. The algorithm locates an optimal solution with less than 1% error and negligible constraint violation in just a few iterations.

Table 11. Optimization result of case study. By using the gray-box approach, the algorithm is able to find a global optimum with $v = 0$.

f^*	f_{actual}	v	N	CPU (hr)
68.0072	68.0072	0	1492	7.4

4.6 Conclusions and Future Perspectives

In this chapter, we propose a data-dependent mixed-integer optimization algorithm for black-/gray- box problems. Unlike existing bb-MINLP algorithms, we do not relax the integrality constraint to construct a surrogate model. Instead, one-hot encoding is used to explicitly handle binary variables. Two surrogate types are considered in this work (ANN and GP) as well as a hybrid model (ANN+GP). These surrogate models are tested and compared with existing bb-MINLP solvers and the results indicate that mixed-integer surrogate models outperform relaxed surrogate models with respect to both solution quality and computational efficiency. We also demonstrate how known constraints can be explicitly incorporated within surrogate-based MINLP formulations through a process synthesis case study to facilitate the search of global optimum. Lastly, we compare different sampling strategies for bb-MINLP optimization and conclude that this has an important effect in the overall performance of the algorithm. The most effective sampling approach ensures that the sampling design is balanced in all combinations of the discrete variables. Our results indicate that when certain constraints are known *a-priori*, these should be directly incorporated within the surrogate-based formulation, because they will significantly limit the feasible search space and will allow the algorithm to focus the exploration and exploitation within feasible subspaces. In addition, we have found that satisfaction of equality constraints is exceptionally difficult in a black-box optimization

setting, and this was quite effectively overcome by the incorporation of a surrogate-based feasibility sampling stage.

The proposed work can be applied to numerous simulation-based problems with both continuous and discrete variables embedded in the simulation. One specific example that is currently being studied is the synthesis of adsorption cycles. Adsorption processes contain different operating steps and cycle configurations that can be represented by binary variables. Decoupling these steps and the associated continuous and binary variables that are embedded in the simulation is not often possible. Using the proposed bb-MINLP algorithm, one can determine the optimal cycle design using input-output data from rigorous adsorption simulation models. Another MINLP case study that is currently being studied is the design of mixed-material, hybrid modular separation systems, for which discrete variables represent the selection of materials and units that are optimal for the separation of different gas mixtures.

Overall, our algorithm shows promise for the solution of MINLP problems with a moderate number of variables and constraints. For applications on problems with significantly more degrees of freedom, the current algorithmic implementation will require improvements to reduce its computational cost by taking advantage of parallel computing, heuristics, and recent exciting advances towards globally optimizing complex NN and GP surrogate formulations. The algorithm is available as an open-source Python library and can be retrieved from http://boukouvala.chbe.gatech.edu/.

CHAPTER 5. SIMULTANEOUS MATERIAL AND PROCESS OPTIMIZATION FOR CARBON CAPTURE

5.1 Introduction

The emission of CO_2 has been recognized as an important environmental issue and one of the major contributing causes of climate change[132-137]. In 2017, the amount of CO_2 emissions in the U.S. totaled 6457 million metric tons[138]. Carbon Capture and Storage (CCS) has been proposed to reduce CO_2 emissions but several existing techniques are currently associated with high cost and large energy consumption[135,139-141]. One promising technology for post-combustion carbon capture is adsorption using solid sorbents due to its relatively high separation efficiency, low energy cost in comparison to absorption-based technologies, and its potential for modularization[142].

During adsorption, separation units containing solid adsorbent sequester CO_2 from flue gas through a dynamic cyclic operation[143,144]. During each cycle, CO_2 is captured and separated from the rest of the mixture; the adsorbent is then regenerated, and the cycle is repeated[143]. Significant research efforts have focused on improving packed-bed adsorbers by overcoming pressure-drop limitations, mitigating adsorption enthalpy, and improving mass transfer to make the operation more cost- and energy-efficient[142,145-148]. Parallel to these efforts, the development of new adsorbents with high working capacities and adsorptive selectivity has also been investigated extensively[149]. Some adsorbents that are commercially available and under development include: activated carbons, zeolites, and metal-organic frameworks (MOFs)[135,149-152].

Recent studies have revealed that the process performance is intricately linked to the choice of an adsorbent, implying that the adsorbent selection and process optimization should be considered simultaneously[140,153-155]. This requires the formulation of a complex simulation-based optimization problem because cyclic adsorption processes are typically described by detailed mathematical models consisting of Partial Differential-Algebraic Equations (PDAEs). Conventional equation-based optimization has been used in [156-158], but it may pose limitations when model complexity is high. Hence, the use of stochastic sampling-based and surrogate-based techniques have been proposed. In [141,154,155,159-161], stochastic optimization algorithms (e.g., Genetic Algorithm (GA)) are used to determine the optimal process operating conditions. However, stochastic algorithms tend to require many simulation evaluations; thus, they may not be suitable when the simulation is computationally expensive. To overcome this limitation, surrogate-based optimization techniques have been proposed[7,13]. Also known as a meta-model, a surrogate model approximates the input-output relationship of a high-fidelity simulation with reduced complexity[1,20]. The surrogate model is subsequently used within algorithms that iteratively sample-fit-optimize surrogates to find optimal solutions.

Several works studying adsorbent selection and process optimization currently exist in the literature. Hasan et al.[140] proposes an adsorbent screening framework using a combined material characterization and process optimization procedure for both PSA and VSA processes. A GP-based gray-box optimization approach is used, and the minimum cost of capture and compression is obtained for the final optimal design satisfying purity and recovery constraints. Khurana and Farooq present a two-stage adsorbent screening framework for a VSA process for carbon capture[154]. They apply a neural network-based

classification model to preliminarily screen adsorbents based on purity-recovery targets; an extensive optimization study is then performed to rank adsorbents and determine the best operating conditions. Leperi et al.[141] performs a full PSA modeling and optimization using GA and introduce a general evaluation metric (GEM) for the screening of MOFs. More recently, Balashankar et al.[153] coupled a GA-based process optimization with a detailed VSA model to develop a zeolite screening framework.

Despite these efforts, challenges still remain for the industrial-scale deployment of adsorptive separations for carbon capture, namely the cost of adsorbents, pressure-drop limitations observed in conventional industrial-size beds, and high adsorption enthalpy[142]. Modular process intensification (PI) offers the opportunity to effectively overcome these technical challenges[162-164]. Modularization may have significant advantages over centralized facilities due to the smaller size of a module, resulting in reduced manufacturing cost and improved transportability. By reducing the size, one can enhance mass and heat transfer and reduce pressure drop, which may lead to 30% energy savings and 20% lower operating costs[162]. In addition, in contrast to centralized facilities that require most of their processes to be built on-site, modules can be mass-produced, leading up to 40% less capital expenditure and shorter module construction and deployment time[162]. These modules can be added or removed depending on plant capacity, allowing flexible deployment. In order to fully exploit these advantages, modules need to be optimally designed so that they can be operated at varying process conditions with the chosen adsorbent.

The purpose of this work is two-fold. First, we propose a technique to design modular CO_2 capture systems for coal-fired power plants using VPSA coupled with

thermally-modulated fiber composite adsorbents. Surrogate-based optimization[2,12,13] is used to determine the optimal adsorbent and process conditions simultaneously. We explore both the Mixed-Integer Nonlinear Programming (MINLP) and Nonlinear Programming (NLP) approaches to surrogate-based optimization. Compared to the brute-force NLP approach, the MINLP approach allows more efficient adsorbent selection and process optimization by using binary variables to represent a set of 75 adsorbents. In the second part, we present a purely machine learning-based classification and regression approaches to gain further insight into adsorbent-process performance through the use of the large amount of data obtained from the simulation-based optimization study. The main novelty of this contribution lies in the use of surrogate-based mixed-integer optimization for simultaneous adsorbent selection and process optimization. We also focus on a novel modular carbon capture system via Vacuum Pressure Swing Adsorption (VPSA) with heat modulating capability. Moreover, this work pushes the scalability boundaries of surrogate-based optimization algorithms with mixed-integer decisions, since these have previously only been applied to low-dimensional integer problems.

This chapter is organized as follows. Section 5.2 provides a background on VPSA modeling as well as an overview of the surrogate-based optimization algorithm. In Section 5.3, adsorbent selection and process optimization are performed using surrogate-based NLP and MINLP algorithms, and a comparison between the two proposed methodologies is presented. Section 5.4 presents the use of dimensionality reduction, classification, and regression techniques to gain further insights into adsorbent-process performance. We end with conclusions and future outlook.

5.2 Background: Process Modeling and Optimization

5.2.1 VPSA Cycle Modeling

5.2.1.1 Process Description

In this work, we consider a VPSA process for a single module with a length of 1m and a diameter of 1/6m. This fixed-bed adsorber is packed with thermally-modulated fiber composites that consist of a porous polymeric matrix spun as a fiber embedded with adsorbent particles (ADS) and phase-change material (PCM). The fraction of the polymer matrix that can be loaded with solids is fixed at a minimum of 25% to guarantee correct fiber manufacturing, while allowing suitable amounts of adsorbent and PCM. In this way, an optimal trade-off between ADS and PCM is achieved in the process design. We present a detailed thermal management study in [165] that addresses this type of contactors based on our recent experimental results[142]. The application of this kind of fiber composite reduces pressure drop in the packed-bed, thus enabling higher gas-velocity operation and better modulation of the heat generated by adsorption, rendering productive cycle operation with optimal adsorbent utilization. Adsorption equilibria are modeled with the extended dual-site Langmuir (DSL) equation as required by the selection of adsorbents applied in the optimization (see Section 2.3). The heats of adsorption for N_2 and CO_2 required for process modeling are estimated for each adsorbent from the corresponding DSL equation parameter values. The 4-step VPSA cycle consists of the following steps: counter-current pressurization (ccPr), high-pressure adsorption (Ad), co-current blow-down (coBd), and counter-current evacuation (ccEv). Figure 25 illustrates this cycle, which allows recovering CO_2 during the evacuation step at high purity. This is enhanced by pressurization with the light-product (LPP) stream, which is rich in N_2.

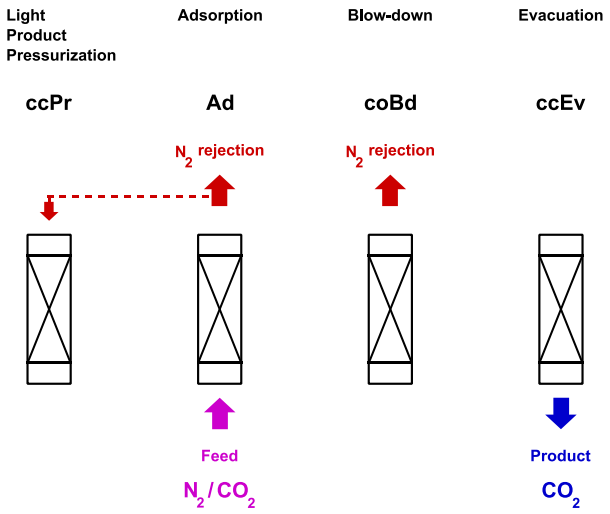


Figure 25. Schematic representation of the 4-step VPSA cycle with LPP.

Extensive studies have been performed on the adequacy of this kind of cycle configuration for CO_2 capture applications – see e.g. [161] and references listed therein. Other cycle designs that consider multi-column operation and column-interaction through pressure equalization steps are also available for this kind of separation but are outside of the scope of this work[166].

5.2.1.2 Mathematical Modeling and Numerical Solution

The detailed full-order dynamic adsorber model for non-isothermal operation consists of a set of partial differential equations (PDEs) in one spatial dimension, which is transformed to an ordinary differential equation (ODE) system in time by application of the first-order upwind discretization scheme (UDS), a finite-volume discretization that is suitable due to its numerical robustness—method of lines (MOL) approach[167]. The time integration of this ODE system proceeds by applying the backward differentiation formulae of Gear[168]. This model is coded and solved entirely in MATLAB[169], applying the 'ode15s'

numerical integration function at default tolerances. We apply suitable boundary conditions (BCs) and initial conditions (ICs) to represent the cyclic operation displayed in Figure 25.

5.2.1.3 Process Optimization

Several degrees of freedom are available to optimize the 4-step VPSA cycle. We select a subset of these decision variables, which have the strongest impact on VPSA process performance. Table 12 lists these variables with the applied bound values.

Table 12. Description of VPSA System Inputs

Input	Operation Bounds	Description
P_{ads}	$2.5 - 20$	Adsorption step pressure (atm)
P_{evac}	$0.01 - 0.5$	Evacuation step pressure (atm)
Q_{feed}	$0.001 - 0.0075$	Feed gas flow rate $(\mathrm{m}^3/\mathrm{s})$
ω_{ads}	$0.15 - 0.5$	Weight-fraction of adsorbent in fiber $\left(\frac{kg\ adsorbent}{kg\ solid\ fiber}\right)$
t_{ads}	$15 - 120$	Adsorption step time (s)

The adsorption and evacuation pressures determine the magnitude of the pressure-swing applied to the process. Feed gas flow rate and adsorption step time control the feed throughput and are therefore critical in the positioning of concentration and temperature fronts along the axial flow direction of the adsorber, once cyclic steady-state (CSS) operation has been attained. The fiber consists of three main components: polymer, adsorbent, and PCM. The weight fraction of adsorbent in the fiber (ω_{ads}) is determined from optimization; the weight fraction of polymer ($\omega_{polymer}$) is fixed to 0.25 to maintain the structural integrity of the fiber. Lastly, the weight fraction of PCM (ω_{PCM}) in the composites establishes the extent to which the heat generated by adsorption is modulated.

The adsorption enthalpy varies between the adsorbents considered, leading to values of PCM content that are specific to each adsorbent and the optimal VPSA operation conditions. The blow-down step pressure is fixed at 1 atm, therefore avoiding the application of two different evacuation pressure levels, which in practice would increase the technical complexity of the applied evacuation system. We select values of the valve-coefficients for the pressure-changing steps (ccPr, coBd & ccEv) that minimize total cycle time. The five decision variables described above are critical in determining the cycle operation and are essential for its optimization as we discuss below.

Table 13. Description of VPSA System Outputs

Outputs	Equations
Productivity	$\frac{mol\ CO_2\ in\ product\ stream}{kg\ adsorbent\ \cdot s}$
Purity	$\frac{mol\ CO_2\ in\ product\ stream}{total\ mol\ in\ product\ stream}$
Recovery	$\frac{mol\ CO_2\ in\ product\ stream}{mol\ CO_2\ Pr\ step,in + mol\ CO_2\ in\ Ad\ step,in}$
Specific Energy Consumption	$\frac{kWh}{tonne\ CO_2}$

In order to evaluate process performance, we use four standard metrics applied in these kinds of separation processes: productivity, product purity, product recovery, and specific energy consumption (Table 13). Additional complexity in the simulation and optimization of VPSA processes results from having to evaluate these metrics at CSS. The transient period before this condition is attained can take several hundreds or even thousands of cycles in some cases, with the associated computational burden. Moreover, a challenging feature encountered by process optimization of cyclic adsorption processes lies in the fact that there exists an inherent trade-off between performance objectives. This is

traditionally addressed by applying multi-objective optimization techniques[170]. The alternative approach we present herein formulates a constrained single-objective optimization task that maximizes the CO_2 productivity of the VPSA cycle, while introducing purity, recovery, and specific energy consumption as constraints to the problem. This is a suitable approach, since there exist well-established specifications adopted worldwide by government agencies that a separation technology for CO_2 capture should fulfill: CO_2 purity of at least 95% and CO_2 recovery of 90%. The energy constraint is needed since the specific energy consumption is directly correlated to the operating cost of the process.

5.2.2 Problem Formulation and Surrogate-Based Optimization for MINLP and NLP

Due to the complexity of the VPSA-cycle simulation, surrogate-based optimization is an attractive alternative to equation-based optimization. In this work, we use a relaxed NN surrogate model to achieve a balance between model accuracy and complexity. While a mixed-integer surrogate model is more accurate than a relaxed model, the use of one-hot encoding increases the dimension of the problem by two as each binary variable is converted to two dummy variables. For this work, we expect that the cost of doubling the dimension would be more significant than the slight loss of model accuracy when relaxing the discrete inputs. Therefore, the relaxed approach was chosen.

In this section, we provide an in-depth explanation of surrogate modeling and optimization for the VPSA process. We consider 75 adsorbents investigated by Khurana & Farooq[154] and originally addressed by Huck et al.[151] for post-combustion CO_2 capture. The

applied dual-site Langmuir adsorption isotherm model in their work[154] consists of 15 parameters specific to each adsorbent.

One main contribution of this work is the use of bb-MINLP algorithm for efficient process optimization and adsorbent selection. Unlike the brute-force approach, where *each* adsorbent is optimized separately using the bb-NLP algorithm, binary variables are used to represent an adsorbent such that *simultaneous* process optimization and adsorbent selection can occur. Even though the bb-MINLP solver initially requires a set of LHD for each adsorbent, further exploration of the material (i.e., binary space) is limited to adsorbents that have promising behavior. Thus, the simultaneous optimization approach will overall exploit process-material relationships and find optimal solutions with fewer samples by avoiding to re-sample for non-promising adsorbents. To test the performance of the bb-MINLP algorithm, we solve this problem using both the bb-MINLP and brute-force bb-NLP approaches. The following optimization formulation (P2) is used to identify the optimal VPSA design that maximizes productivity subject to 95% purity and 90% recovery constraints with an upper bound on the specific energy consumption of 2000 $kWh/tonne\ CO_2$:

$$\begin{aligned} &max \text{ Productivity} = \\ &f\left(P_{high}, P_{evac}, t_{ads}, \omega_{ads}, Q_{feed}, y_{ads}^{k}\right) \\ &s.t. \text{ Purity} = g_1(P_{high}, P_{evac}, t_{ads}, \omega_{ads}, Q_{feed}, y_{ads}^{k}) \geq 0.95 \\ &\text{Recovery} = g_2(P_{high}, P_{evac}, t_{ads}, \omega_{ads}, Q_{feed}, y_{ads}^{k}) \geq 0.9 \end{aligned} \tag{21}$$

$$\text{Specific Energy Consumption} = g_3(P_{high}, P_{evac}, t_{ads}, \omega_{ads}, Q_{feed}, y_{ads}^k) \leq 2000$$

$$2.5 \leq P_{high} \leq 20, 0.01 \leq P_{evac} \leq 0.5, 15 \leq t_{ads} \leq 120,$$

$$0.15 \leq \omega_{ads} \leq 0.5, 0.001 \leq Q_{feed} \leq 0.0075$$

$$y_{ads}^k \in \{0,1\}, \sum_{k \in adsorbents} y_{ads}^k = 1, k = 1, \ldots, K$$

Binary variables y_{ads}^k represent an adsorbent k, where $k \in$ set of 75 adsorbents (i.e., $K = 75$). If $y_{ads}^{k'} = 1$, then adsorbent k' is selected. An additional constraint $\sum_k y_{ads}^k = 1$ is needed to allow the selection of a single adsorbent. For the brute-force approach, k= ∅; hence, the problem reduces to NLP. The structure of the neural network is shown in Figure 26. It has 80 input nodes, a single hidden layer, and 4 output nodes.

5.3 Results and Discussion: Adsorbent Selection and Process Optimization

In this section, we present the optimization results for adsorbent selection and process optimization using both bb-NLP and bb-MINLP approaches. The bb-NLP approach performs a separate optimization for *each* adsorbent; the bb-MINLP approach conducts a single optimization for *simultaneous* adsorbent selection and process optimization. These results are compared with respect to computational efficiency and solution accuracy.

The optimization framework is written in Python. The VPSA simulation is coded in Matlab2018b[169], and a Python-Matlab interface is applied. Since the simulation is

computationally expensive, 5 processors are used. A neural network surrogate model is constructed with the Python module 'scikit-learn', and the optimization is performed using GAMS. At each iteration, all local and global optimal solutions are collected using DICOPT[129] (or CONOPT[129] for NLP) and BARON[83] solvers, respectively. The maximum allowed computation time is 50 hours. The optimization is repeated 3 times, and the best result is reported. Table 14 summarizes the main difference between MINLP and NLP strategies.

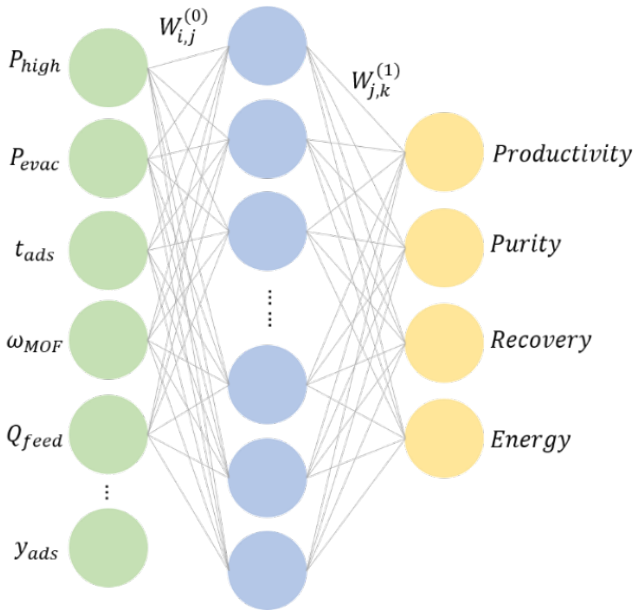


Figure 26. Illustration of a neural network with an input, one hidden, and an output layer

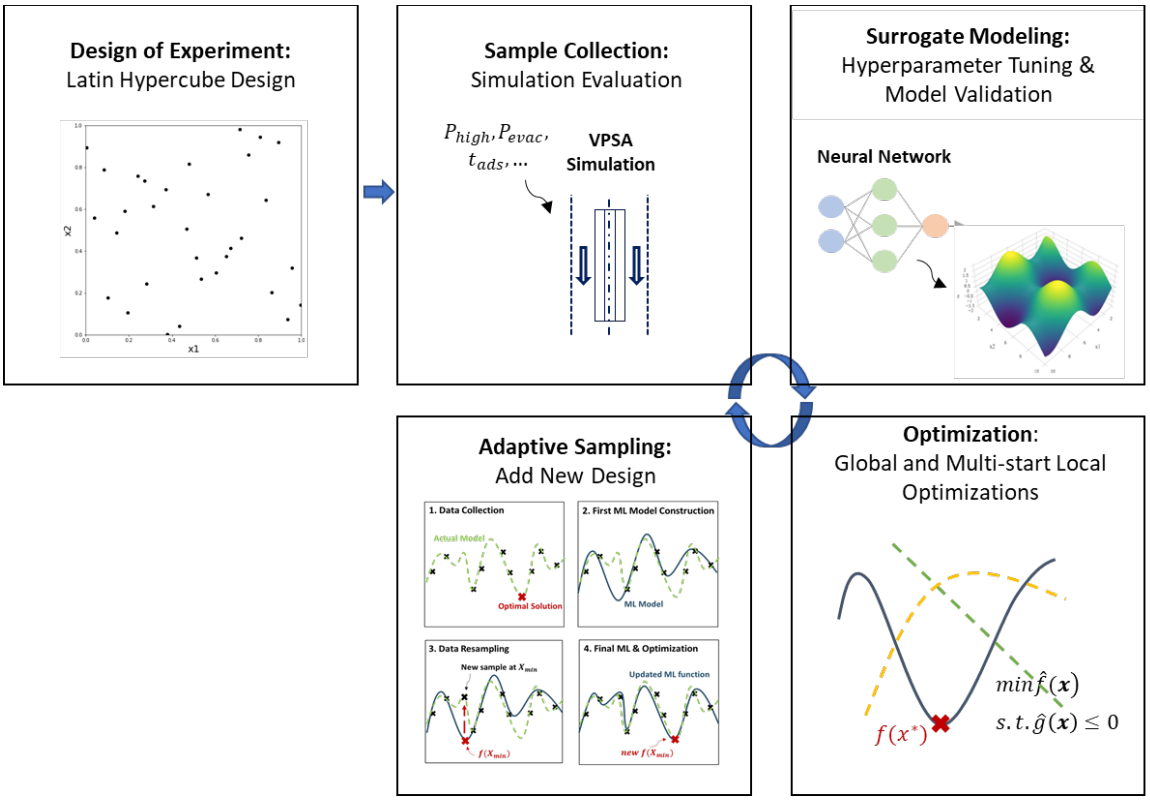


Figure 27. Overview of surrogate-based optimization for VPSA process optimization. These steps are repeated until one of the convergence criteria is met.

Table 14. Comparison of MINLP and NLP strategies

Approach	# Problems	Dimension	LHD size	ANN structure	Optimization solvers
bb-MINLP	1	80	3825	80-58-4	Baron, Dicopt
bb-NLP	75	5	51	5-8-4	Baron, Conopt

5.3.1 bb-NLP Optimization: Brute-Force Approach

Process optimization is performed separately for all 75 adsorbents using a brute-force NLP approach. For each adsorbent, an initial LHD of size 51 ($10I + 1$, where $I = 5$) is created. A single layer neural network is constructed with 5 input, 8 hidden, and 4 output nodes to predict 4 process performance metrics (i.e., productivity, purity, recovery, and

specific energy consumption). Table 15 shows the statistical summary of the output data generated from the initial LHD for all 75 adsorbents. We can observe that there exists a huge variability in the dataset, where most output data points are not feasible. This implies that the feasible region of the search space is very small and finding an optimal solution is difficult.

Table 15. Statistical summary of output data generated from initial LHD

Output	Productivity (mol/kg·s)	Purity	Recovery	Energy $(\frac{\text{kWh}}{\text{tonne CO}_2})$
Mean	0.004387	0.6151	0.0963	45961
Stdev	0.002577	0.1003	0.1097	52972

Table 16 shows the optimal process conditions and the performance of all feasible adsorbents ranked in the order of decreasing productivity. Figure 28 shows the scatterplots of optimal productivity vs. 5 process inputs of all 75 adsorbents. If an adsorbent violates one or more constraints, it is considered "infeasible"; if all constraints are satisfied, an adsorbent is "feasible". From the optimization result of 75 adsorbents (Table 16 and Figure 28), we can notice a few trends. First, in terms of constraint violation, the purity constraint was the most difficult one to satisfy (i.e., satisfied only 33% of the times), followed by recovery (48%) and energy (88%). For the operating pressure, feasible adsorbents tend to cluster at high P_{high} and low P_{evac}. Operating the VPSA cycle at a higher P_{high} enables higher concentration of CO_2 in the feed gas during the adsorption step (Ad) and an increase in the pressure-swing. The cycle operation at lower evacuation pressure, P_{evac} , also contributes to increase the pressure-swing, therefore improving CO_2 recovery and allowing better adsorbent regeneration during the counter-current evacuation step (ccEv), while

sustaining the CO_2 product purity target. For the feed gas flow rate, feasible adsorbents tend to cluster at lower Q_{feed}, values. Q_{feed} should be such that enough contact time is provided for the gas uptake to occur; however, high values of feed gas flow rate induce losses of CO_2 at the light-product end of the fixed-bed during the co-current blow-down step (coBd), impacting negatively CO_2 recovery. In the case of the adsorption time, t_{ads}, both feasible and infeasible adsorbents tend to converge to the lower bound of t_{ads}, which is explained by the fact that enabling shorter cycle times translates to better productivity results. While the overall trend is also less obvious for the content of adsorbent in the fiber composite, ω_{ads} , when all 75 adsorbents are considered, we found an interesting trend among the feasible adsorbents. As shown in Table 16, the productivity of an adsorbent is inversely proportional to ω_{ads}. In fact, two adsorbents that significantly outperform the rest of the adsorbents – Zn-MOF-74 and UTSA-16 – converge to the lower range of ω_{ads}, and as the optimal productivity decreases, ω_{ads} increases. Since the fiber packing fraction is fixed, decreasing adsorbent content in the fiber composite is equivalent to increasing its PCM content. This achieves better temperature modulation[165]. Hence, for adsorbents capable of high CO_2 uptake, the presence of PCM in the fiber composites can further enhance adsorption by effectively managing the heat generated.

Table 16. Optimal process conditions for 26 feasible adsorbents that meet 95-90 Purity-Recovery constraints. The adsorbents are listed in the order of decreasing productivity.

	Optimal Process Conditions					Performance Measurements			
	P_{high} (atm)	P_{evac} (atm)	Q_{feed} (1e-3 m3/s)	t_{ads} (s)	ω_{ads}	Prod (1e-2 mol/kg·s)	Purity (%)	Recovery (%)	Energy $(\frac{kWh}{tonne\ CO_2})$
Zn-MOF-74	20	0.069	1.78	17.85	0.150	4.91	95.10	90.06	860
UTSA-16	19.92	0.088	1.05	25.29	0.150	4.25	95.57	90.22	828
MgX	20	0.036	1.03	24.88	0.150	2.66	95.11	91.18	857
Co-MOF-74	20	0.029	1.05	38.16	0.150	2.43	95.11	90.06	876
ZIF-39-DIA	19.83	0.072	1.38	15.42	0.331	1.96	95.61	90.06	881
NAB	20	0.052	1.02	15.00	0.280	1.85	95.17	90.04	904
h8155527	20	0.074	1.22	15.00	0.348	1.74	95.05	92.51	833
ZIF-82	20	0.033	1.00	30.01	0.291	1.59	95.06	90.09	909
13x	13.46	0.054	1.20	29.17	0.293	1.56	95.13	90.39	708
HMOF-992	20	0.031	1.00	26.74	0.298	1.56	95.16	90.12	950
ZIF-69	20	0.035	1.06	27.25	0.341	1.44	95.05	90.24	921
ZIF-116-MER	20	0.035	1.00	42.49	0.360	1.38	95.40	90.01	929
ZIF-78	20	0.042	1.00	34.69	0.362	1.31	95.00	90.20	875
CaX	20	0.025	1.00	36.01	0.226	1.25	95.05	90.00	877
NaA	20	0.078	1.00	32.90	0.478	1.25	95.52	91.08	829
Al-X	20	0.035	1.00	31.58	0.313	1.22	95.03	91.01	854
CuBTTri	20	0.040	1.02	68.39	0.470	1.21	95.76	90.35	909
ZIF-36-FRL	18.46	0.047	1.00	26.47	0.365	1.16	95.01	90.09	820
CuBTC	20	0.010	1.00	42.32	0.231	1.15	95.06	95.08	995
Na-X	19.88	0.035	1.06	36.51	0.408	0.96	95.40	90.11	865
Mg-X	15.82	0.051	1.00	51.58	0.500	0.88	95.39	90.04	763
h8124767	8.89	0.032	2.20	39.71	0.400	0.82	95.18	91.45	572
ZIF-68	18.62	0.017	1.00	36.24	0.445	0.79	95.03	92.54	932
ZIF-81	19.24	0.018	1.00	40.50	0.500	0.66	95.23	90.02	936
h8291835	19.83	0.015	1.13	18.12	0.500	0.65	95.10	90.21	996
NaX	13.68	0.029	1.00	71.78	0.500	0.57	95.02	92.72	710

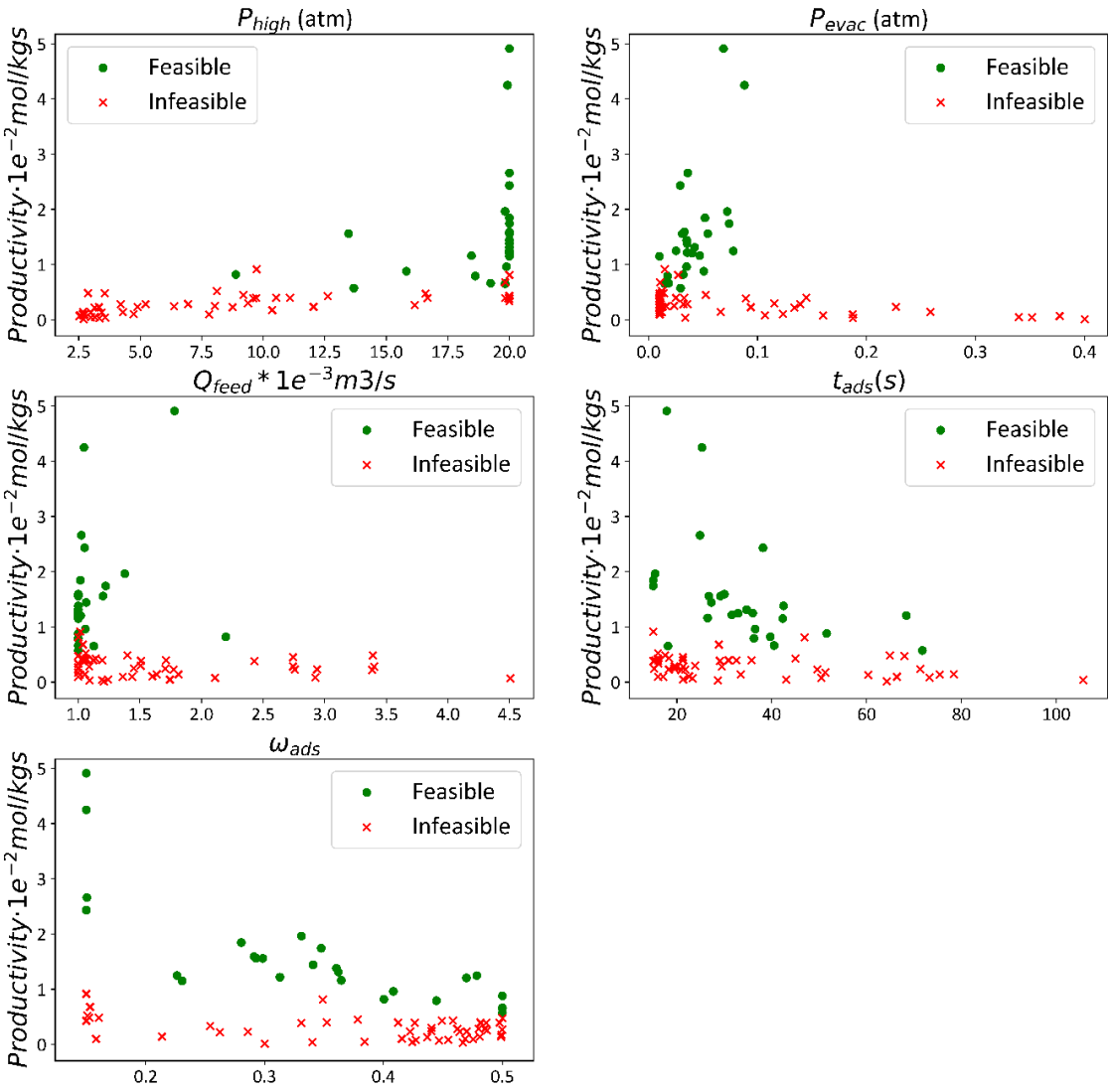


Figure 28. Optimal productivity is plotted with 5 process inputs (P_{high}, P_{evac}, Q_{feed}, t_{ads}, and ω_{ads}). Feasible adsorbents (green) satisfy the required purity-recovery-energy constraints.

5.3.2 bb-MINLP Optimization: Simultaneous Approach

Previously, we used a surrogate-based NLP algorithm to conduct process optimization of 75 adsorbents separately. This optimal set of results provides a valuable insight into adsorbent performance when coupled with data analytics techniques, which we explore in Section 4. The brute-force approach with $K = 75$ optimization problems to be

solved to convergence individually is computationally expensive. However, if we treat this as a bb-MINLP problem, binary variables, y_{ads}^{k} , are used to represent 75 adsorbents, and (P2) is solved as a single problem. To create a balanced initial sample set, an initial LHD design with a size of $10I + 1$ is constructed for each adsorbent k. These LHD sets are then combined to construct a final sample set with $K(10I + 1)$ points. With y_{ads}^{k}, the neural network now has $K + 5$ input nodes, 4 output nodes, and one hidden layer. The MINLP search step preliminarily determines the most promising adsorbent y_{ads}^{k}, while the NLP search is then performed for the most promising adsorbent to further refine the solution with respect to the design variables: $P_{high}, P_{evac}, t_{ads}, Q_{feed}, \omega_{ads}$. Table 17 shows the optimization result of the simultaneous bb-MINLP approach. The optimal adsorbent is Zn-MOF-74 with an optimal productivity of $0.0481\ mol/kg \cdot s$. Note that for all the bb-NLP and bb-MINLP solutions, the final purity and recovery values converged slightly above 95% and 90%, respectively. This is due to small approximation errors of regression models that lead to solutions that do not fall exactly on the feasibility boundaries.

Table 17. Simultaneous Approach Optimization Result

	Optimal Process Conditions					**Performance Measurements**			
y_{ads}	P_{high} **(atm)**	P_{evac} **(atm)**	Q_{feed} **(1e-3 m3/s)**	t_{ads} **(s)**	ω_{ads}	**Prod (1e-2 mol/kg·s)**	**Purity (%)**	**Recovery (%)**	**Energy (kWh/tonne CO_2)**
Zn-MOF-74	19.83	0.064	1.23	24.73	0.15	4.81	95.97	90.33	853

5.3.3 Comparison between bb- NLP and bb-MINLP Approaches

A comprehensive comparison of bb-NLP and bb-MINLP approaches with respect to sampling and computational requirements is presented in this section. The bb-NLP

approach is computationally expensive, solving 75 individual surrogate-based optimization tasks. While this computational cost can be alleviated to some extent by using parallel computing, this computational resource may not always be available. The bb-MINLP approach is computationally more efficient since it only requires a single surrogate-based optimization to be performed and exploits the material-process search space efficiently by avoiding re-sampling of non-promising materials.

Table 18. Comparison of computational cost between bb-NLP and bb-MINLP approaches. For bb-NLP, both the total and average computation time are reported to sequentially optimize all 75 adsorbents and a single adsorbent, respectively.

	# samples	Sampling (hr)	Modeling (hr)	Optimization (hr)	Total
bb-NLP (total)	29335	893.76	9.53	5.76	909.04
bb-NLP (average)	391	11.92	0.13	0.08	12.12
bb-MINLP	5447	13.41	1.72	11.46	26.59

Table 18 lists the sampling requirement and computation time of bb-NLP and bb-MINLP approaches. For the bb-NLP approach, we first report the total computation time for all 75 adsorbents, assuming that the optimization is performed sequentially. The computation times of each stage of this optimization strategy — sampling, model fitting, and optimization — are shown. As expected, the bb-MINLP approach requires 97% times less computation time than that of the bb-NLP approach. When the computation time of the three stages is compared, the most computationally expensive stage consists of collecting samples from the VPSA simulation. In fact, most of the computation time is spent during the sample collection stage, and the model fitting and optimization stages are significantly faster.

In Table 18, the average computation time per adsorbent is also reported for bb-NLP. Looking at the average allows us to compare the computational efficiency of the algorithm for a single optimization. On average, the bb-NLP approach requires a smaller number of samples and computation time for a single adsorbent. This is expected since the bb-NLP approach constitutes a simpler problem with 5 input variables only. The bb-MINLP approach requires more samples and computation time because it is a more challenging problem with 80 input variables. In particular, the optimization stage contributes to about 43% of the total computation time, while that of the brute-force approach is less than 1%. This is also expected since the deterministic optimization of an MINLP problem is more difficult than that of the NLP problem. Nevertheless, when the total computation requirement is considered, we can conclude that the bb-MINLP approach is computationally more efficient overall.

In terms of solution accuracy, both the bb-MINLP and bb-NLP approaches identified Zn-MOF-74 as the best adsorbent with productivity $0.0481\ mol/kg \cdot s$ and $0.0491\ mol/kg \cdot s$, respectively. While the MINLP search stage correctly identifies the optimal adsorbent, the NLP stage can lead to a slightly different optimal result, which is typical when applying surrogate-based approaches due to stochasticity caused by different sampling locations and model training.

5.4 Analysis of Adsorbent-Process Interaction using Data Analytics and Machine Learning

In Section 3, we collect all results obtained from the previous analysis (optimal process conditions) for all 75 adsorbents. Each adsorbent possesses a particular set of dual-

site Langmuir equilibrium parameters. Our aim here is to perform some analysis on the merged process-material data and gain further insights into the correlations between process conditions and adsorbent. We first compute a correlation matrix to observe how all process features and isotherm features are correlated. We then perform Principal Component Analysis (PCA) to handle highly correlated variables and observe the importance of process and adsorbent features. Finally, we construct machine learning-based classification and regression models that allow us to predict adsorbent feasibility and performance.

5.4.1 Correlation Matrix of Isotherm and Process Features

A correlation matrix is constructed to observe how the input variables are correlated. Investigating the correlation between independent variables is an important stage in machine learning since multi-collinearity can potentially create difficulty in estimating model parameters[171].

Both the adsorption isotherm equation features (i.e., 15 DSL parameters) and process features (i.e., optimal operating conditions determined in Section 5.3.1) of all 75 adsorbents are included in this analysis. In Figure 30, labels $qsat_{11}$, $qsat_{12}$, b_{11}, b_{12}, B_{11}, B_{12}, ΔH_1 correspond to DSL parameters and isosteric heat of adsorption for N_2; labels $qsat_{21}$, $qsat_{22}$, b_{21}, b_{22}, B_{21}, B_{22}, ΔH_2 correspond to DSL parameters and isosteric heat of adsorption for CO_2. Note that $qsat_{ij}$ is a saturation capacity for site i for species j (mol/m^3), b_{ij} is a Langmuir constant for species i (m^3/mol), B_{ij} represents the change in the internal energy due to adsorption for species i (kJ/mol), and density denotes the adsorbent density (kg/m^3).

Figure 30 shows a correlation matrix generated from 75 adsorbents. As expected, isotherm parameters tend to be highly correlated (e.g., $qsat_{11}$ and $qsat_{12}$) since these parameters are typically estimated by nonlinear fitting of either experimentally collected data points or data generated in-silico[154]. Thus, we can observe higher correlation values in the upper left quadrant of the correlation matrix. On the other hand, operation features tend to be less correlated, because these are design variables that are optimized. Some moderate correlation is still observed among P_{high}, P_{evac} and Q_{feed} as observed in the lower right quadrant of the correlation matrix.

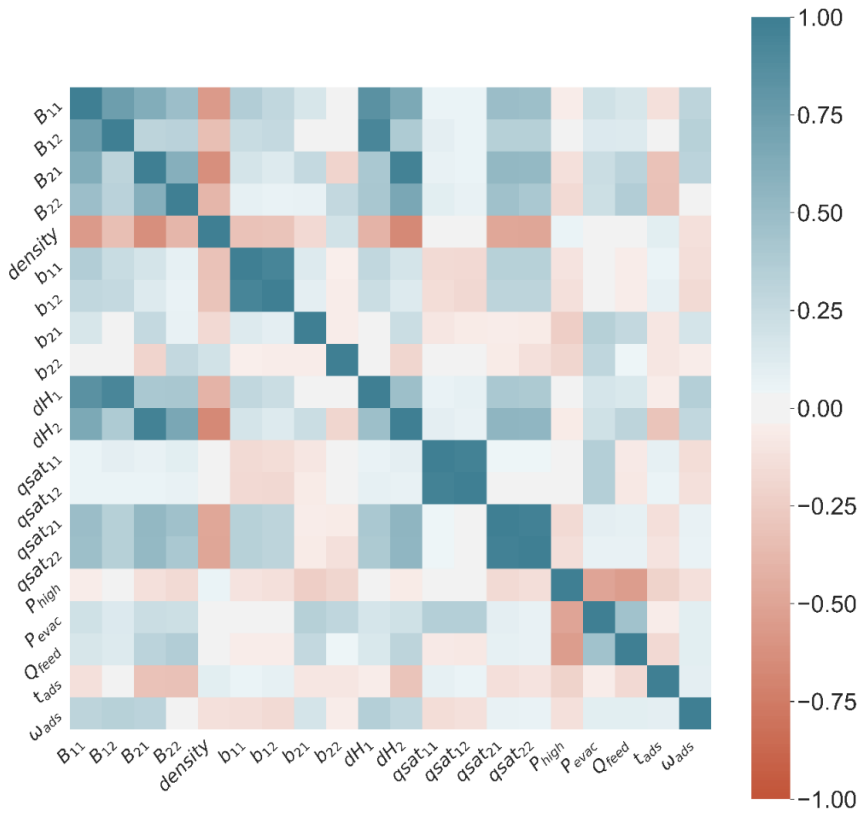


Figure 29. Correlation matrix of 15 isotherm features and 5 process features. Darker colors (both blue and red) represent higher correlation. The decision variables are also included in the matrix.

5.4.2 PCA Results

PCA is a dimensionality reduction technique used for data visualization and handling data multi-collinearity[172]. To observe any pattern or clustering among feasible and infeasible adsorbents, we combine 15 adsorbent features (i.e., isotherm parameters) and 5 optimal process features for 75 adsorbents and perform linear PCA. Note that we have tested several nonlinear kernels (e.g., polynomial, radial basis function, sigmoid, and cosine), but they did not improve the visualization of feasible/infeasible adsorbents. Due to space limitations, we did not include the results here. Figure 30 shows the cumulative explained variance vs. the number of principal components (PCs). With just 6 PCs, we can explain about 80% of the data variance, which means that we can potentially decrease the dimension of the problem from 20 to 6 and still accurately capture most of the data variance.

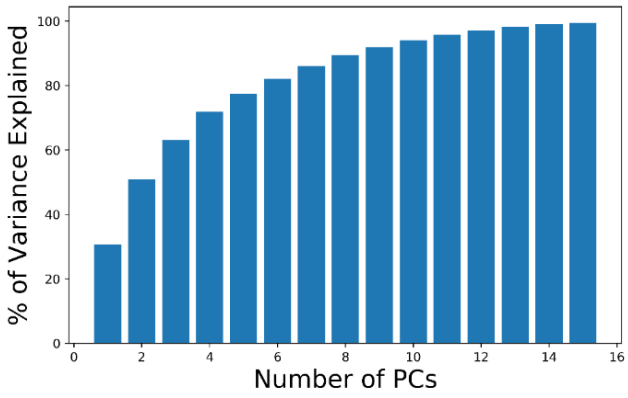


Figure 30. Percentage of variance explained vs. the number of PCs

For linear PCA, each PC is represented as a linear combination of original features: $PC_p = \sum_{f \in F} w_f(x - \mu_f)$, where p represents the selected PC and F represents all 20 isotherm and process features. We can analyze the importance of each feature by analyzing

its weight w_f, where a feature with larger $|w_f|$ is considered more important. To compare the importance of adsorbent and process features for the first six PCs, we computed their percentage contribution, expressed by: $\%\, Contribution = 100 \cdot \frac{\Sigma_{f \in F} |w_f|}{\Sigma_{f \in F} |w_f|}$, where F' represents a set of either adsorbent or process features. Figure 31 shows the computed feature contribution.

For the first PC, adsorbent features explain about 70% of data variance; for the second PC, adsorbent features explain about 50% of data variance. While adsorbent features seem to contribute slightly more to the PCs than the process features, it is difficult to conclude the existence of a dominant feature. Figure 31 supports our claim that both adsorbent and process features contribute to the overall data variance and are both important for the VPSA process design.

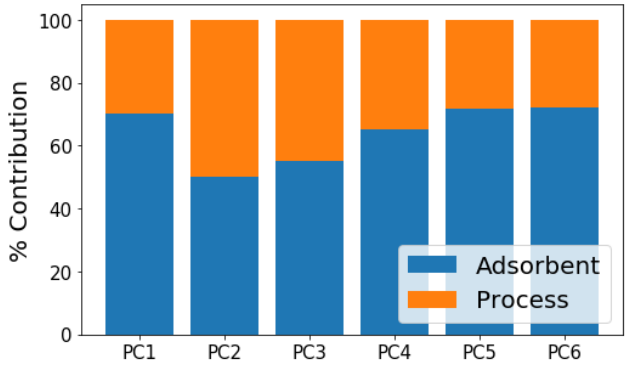


Figure 31. Feature % contribution of the first 6 PCs.

Finally, the first two PCs are plotted to visualize the data in a 2-dimensional space and observe any pattern among feasible and infeasible adsorbents. From Figure 32, we can observe some clustering of feasible adsorbents in the lower left quadrant of the PC space.

In fact, we have observed that the performance of an adsorbent improves as we approach the lower left quadrant in the PC space.

5.4.3 Support Vector Machine (SVM) Classification Model for Adsorbent Feasibility

Using the PCs obtained in the previous section, we construct a classification model that predicts adsorbent feasibility, assuming that we are operating an adsorbent at optimal operating conditions. As shown in Figure 32, the feasible and infeasible adsorbents are nearly linearly separable in the 2-dimensional PC space. To exploit this trend, we construct a linear SVM classification model to classify adsorbent feasibility using 2 PCs. A linear SVM model seeks to find a linear hyperplane that can separate two classes of points and it has a regularization hyperparameter C, which can be tuned using a grid search[94]. For model training, 80% of the data is used with 5-fold cross-validation, and 20% of the data set is set aside to test how well the model generalizes to a new set of data. The resulting linear SVM model is also shown in Figure 32. For the training set, the SVM model accuracy is 82%, and this error results from the fact that the points are not perfectly linearly separable. For the test set, the model accuracy is 100%, which implies good generalizability. While we can generally improve the model accuracy by increasing the number of PCs or considering nonlinear classification techniques (e.g., rbf kernel), neither of these two approaches significantly improved our results (i.e., nonlinear techniques led to ~5% model accuracy increase). Hence, a linear SVM model with 2 PCs is chosen for its simplicity and the ease of visualization.

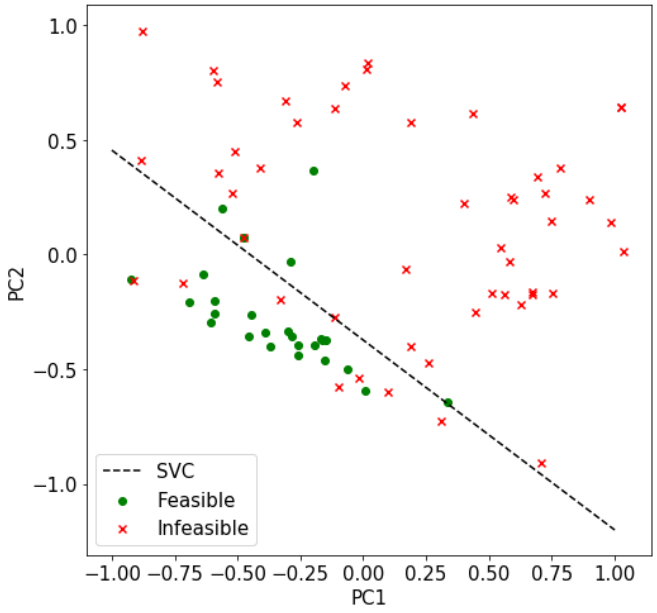


Figure 32. PCA is performed and the first two PCs are plotted, where feasible adsorbents that satisfy all constraints are indicated by green dots. The decision boundary found via support vector classification is also displayed.

5.4.4 Adsorbent Performance Prediction Model

In the previous section, we constructed an SVM-based adsorbent feasibility classification model using 2 PCs. While this model is sufficient when one is interested in determining whether an adsorbent is feasible or not, it is not sufficient to provide detailed information on the performance of the VPSA system in terms of product productivity, purity, recovery, and specific energy consumption. Thus, we construct a neural network-based adsorbent performance prediction model to predict the VPSA system performance given isotherm parameters and the optimal operating conditions.

Previously in Section 4.1, we have determined that isotherm features (i.e., dual-site Langmuir isotherm parameters) tend to be highly correlated, while process features exhibit less correlation. Hence, we perform linear PCA on just the isotherm features to reduce the dimension from 15 to the selected number of PCs and handle existing correlations between isotherm parameters. These isotherm-based PCs are then combined with optimal process features to construct a prediction model. Figure 33 summarizes the result of linear PCA. Even with just 2 PCs, we can capture more than 60% of data variance; with 5 PCs, we can capture ~80% of data variance. When the first two PCs are plotted, we did not observe any pattern among feasible and infeasible adsorbents. This further enhances our previous claim that both operation and adsorbent features are important in predicting the feasibility of an adsorbent; thus, both features must be considered.

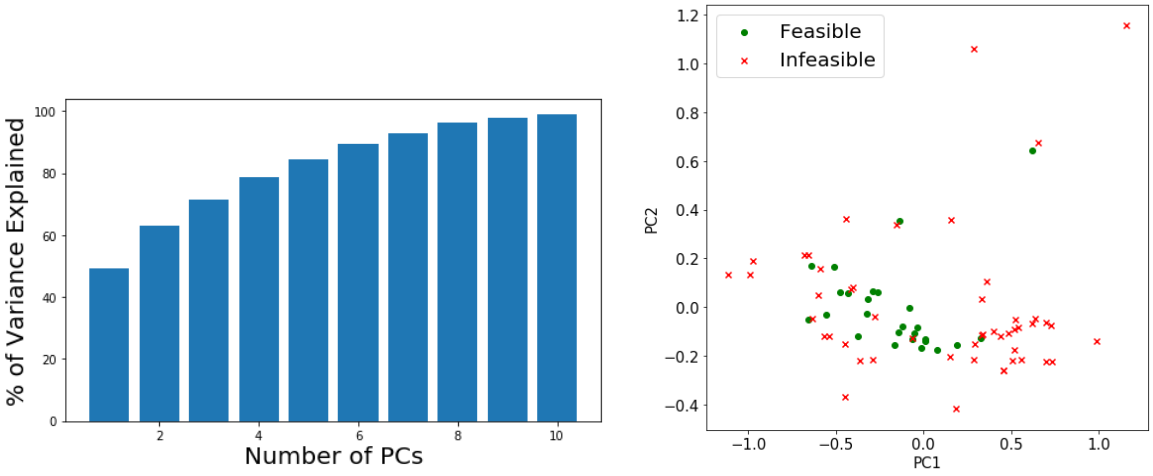


Figure 33. PCA on only isotherm parameters (PC_{ads}): (a) Percentage of variance explained vs. the number of PCs, and (b) 2-d representation of PC space

After reducing the dimensionality of isotherm features using PCA, the first two PCs are combined with the optimal process features, which results in 7 inputs (i.e., $PC_{ads,1}$, $PC_{ads,2}$ P_{high}, P_{evac}, Q_{feed}, t_{ads}, and ω_{ads}). These inputs are then used to train 4 neural network models that can predict product productivity, purity, recovery, and energy

consumption. For the selection of neural network hyperparameters, a grid search is used with 5-fold cross-validation to select the best set of hyperparameters from: hyperbolic tangent (tanh) or rectified linear unit (relu); the number of hidden layers (from 1 to 4 layers); the number of nodes per hidden layer (varied from 5 to 30 per layer). The productivity, purity, and recovery models are trained using the best result out of three runs for all 75 adsorbents. We found that the most challenging output to predict was energy, and we hypothesize that this is because the energy constraint is relatively easier to satisfy and is often non-active in the optimal solution. This creates some more variation in the optimal energy values. Nevertheless, when using all optimal results (i.e., 225 optimal data points from all three runs of 75 adsorbents) to construct an energy consumption neural network model, we can generate a model with high accuracy. Table 19 shows the best hyperparameters determined from grid search, Figure 34 shows the parity plot of four neural network models, and Table 20 shows the train-set and test-set goodness-of-fit given by R^2. The normalized RMSE is also reported to facilitate the comparison between models with different observed ranges. We have observed that increasing the number of PCs does not significantly improve the model. Therefore, applying only 2 PCs is a good compromise that balances model accuracy and complexity.

Table 19. Best neural network model determined from hyperparameter grid search

	Activation	**# hidden layers**	**# nodes per hidden layer**
Productivity	relu	3	5-9-7
Purity	tanh	3	7-9-7
Recovery	relu	4	7-7-5-5
Energy Consumption	tanh	2	5-9

As illustrated in Figure 34 and Table 20, we have successfully constructed accurate VPSA performance prediction models with average $\bar{R}^2_{train} = 0.98$ and $\bar{R}^2_{test} = 0.86$. These models can be used to preliminarily evaluate the performance of an adsorbent using isotherm and process features.

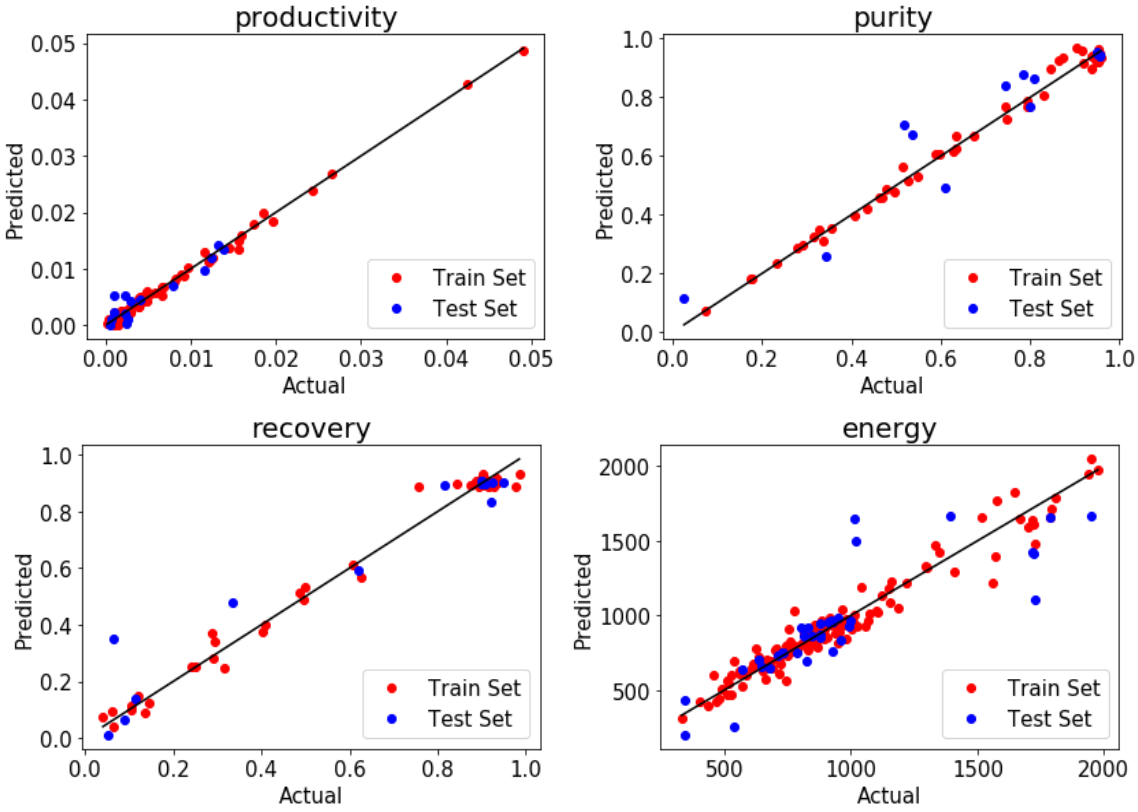


Figure 34. Parity plot of productivity, purity, recovery, and energy neural network models. The y-axis is the predicted value from a neural network model, and the x-axis is the actual simulation output. Note that the energy model is trained using all optimal results.

Table 20. Train set and Test set R^2 and NRMSE for productivity, purity, recovery, and energy neural networks.

	Productivity	Purity	Recovery	Energy
R^2_{train}	0.99	0.99	0.99	0.94
R^2_{test}	0.88	0.90	0.94	0.70
$NRMSE_{train}$	1.12e-5	6.23e-4	1.15e-3	4.14
$NRMSE_{test}$	2.20e-4	7.52e-3	9.03e-3	28.50

5.5 Conclusions and Future Perspectives

In this work, we propose a surrogate-based optimization approach for the design of modular VPSA systems for post-combustion CO_2 capture. To achieve optimal performance, we consider both the adsorbent selection and process operation conditions to design a modular VPSA system. We investigate two different approaches to formulate and solve the optimization problem: 1) the bb-NLP approach, where the process optimization is performed for each adsorbent, and 2) the bb-MINLP approach, where the adsorbent selection and process optimization are performed simultaneously. When all 75 adsorbents are compared, the bb-MINLP approach is more efficient with respect to sampling and computational requirements.

In addition to the design of a module, we also demonstrate how machine learning classification and regression techniques can be applied to identify feasible adsorbents and predict the performance from a purely data-driven perspective. In the 2-dimensional PC space, a clustering of feasible adsorbents has been observed, and a linear SVM classification is developed. Finally, we construct a neural network-based performance prediction model to predict four outputs of the VPSA simulation (i.e., purity, recovery, energy consumption, and productivity). These classification and neural network models provide valuable preliminary insights into different adsorbents and process performance.

CHAPTER 6. MODULE SIZING, COSTING, AND ECONOMIC ANALYSIS FOR POWER PLANTS

6.1 Introduction

Power plants in the U.S. are one of the major contributing sources of CO_2. In 2019, electric power plants emitted 1,619 million metric tons of CO_2, which corresponds to 32% of total U.S. energy-related CO_2 emissions[173]. Currently, three major sources of energy for electricity generation are fossil fuels (e.g., coal, natural gas, and petroleum), nuclear energy, and renewable energy sources. While the size of the renewable energy market is slowly growing, fossil fuel is still a leading source of electricity generation. In particular, coal power plants, which produced 23% of energy in the U.S. in 2019, emitted 973 million metric tons of CO_2; natural gas was used to generate 38% of electricity, emitting 619 million metric tons of CO_2(Table 21).

Table 21. CO_2 emissions by the electric power sector in the U.S. (2019)[173]

Source	Million metric tons	Share of sector total
Coal	973	60%
Natural gas	619	38%
Petroleum	16	1%
Other	11	1%
Total	1619	100%

Carbon capture is one promising solution to mitigate carbon emission. In the previous chapter, we have proposed the use of surrogate-based MINLP optimization for simultaneous adsorbent selection and process optimization. A set of promising adsorbents and the optimal process operating conditions were identified for a VPSA module. In this final chapter, we focus on two adsorbents that were previously determined to exhibit good

performance: zeolite 13x and Zn-MOF-74. These two adsorbents were selected because they have opposing characteristics: Zn-MOF-74 had the highest productivity among all feasible adsorbents but it is much more expensive than zeolite 13x; zeolite 13x, while much cheaper and more readily available, is less efficient than Zn-MOF-74. Using these two adsorbents, we expand our previous work by exploring the feasibility of deploying VPSA modules to power plants in the U.S. First, we improve the optimal module design by incorporating module dimensions (length and diameter) as additional design variables to optimization. Both coal and natural gas sources are explored. We then conduct an economic analysis to calculate the capture cost using the power plant data published by the U.S. Energy Information Administration (EIA)[174]. The dataset contains information on electricity generation and CO_2 emission, which will be used in our economic analysis. As some power plants are retiring after 2020, the dataset is reconciled with a list of operating power plants beyond 2020. All data manipulation is performed in Python.

6.2 Process Optimization for Power Plants

In this chapter, we explore two major sources of CO_2 emission: coal and natural gas power plants. In 2019, 1899 natural gas power plants generated almost 40% of electricity in the U.S., while 308 coal power plants generated 23% of electricity[173]. Since the number of coal power plants is declining and that of natural gas has constantly inclined for the past decade, the possibility of including natural gas power plants in the study is explored. Flue gas emitted from coal power plants and natural gas power plants have different concentrations of CO_2: ~14% for coal power plants and ~4% for natural gas power plants[133]. Optimization is performed to determine the optimal module design with 7 design variables: adsorption pressure (P_{ads}), evacuation pressure (P_{evac}), adsorption time (t_{ads}),

adsorbent loading in the fiber (ω_{ads}), volumetric flowrate of the feed (Q_{feed}), module length (L_{module}), and module diameter (D_{module}). Previously in chapter 4, the module length and diameter were fixed at 1m and 1/6m, respectively. In this chapter, module length and diameter are included as design variables to further enhance the performance of a VPSA module.

6.2.1 Process Optimization for Coal Power Plants

One major factor that affects the capital cost of a modular facility is n_{feed}, which is the molar flow rate of the flue gas per module. A module with increased product productivity has a higher n_{feed}. Thus, it can process more flue gas with fewer modules, reducing the initial capital investment. On the other hand, the operating cost is significantly affected by the energy requirement. Decreasing the energy requirement involves decreasing the adsorption pressure, which leads to decreased productivity. To observe how these opposing characteristics affect the overall cost of a modular facility, we study 2 different types of modules in this section.

Previously, module designs were obtained for Zn-MOF-74 and zeolite 13x by maximizing product productivity as the objective. While maximizing productivity allows us to obtain an efficient design that can maximize the module capacity, these modules require high energy consumption (~800 kW/ton CO_2 captured). This may translate to lower capital investment but higher operating expenses. To study the tradeoff between capital and operating costs, two additional module designs are included in this study by solving the following optimization problem:

$$min\, Specific\ energy\ consumption$$

$$s.t.\, Productivity \geq 0.001$$

$$Purity \geq 0.95 \tag{22}$$

$$Recovery \geq 0.9$$

Four module designs are obtained using the bb-NLP algorithm and shown in Table 22.

Table 22. Final module design for coal power plants

	Module Type 1 (max productivity)		**Module Type 2 (min energy consumption)**	
	Zn-MOF-74	**13x**	**Zn-MOF-74**	**13x**
$\boldsymbol{P_{ads}}$ **(atm)**	20	15.19	5.34	7.43
$\boldsymbol{P_{evac}}$ **(atm)**	0.069	0.025	0.011	0.015
$\boldsymbol{\omega_{ads}}$	0.15	0.201	0.168	0.249
$\boldsymbol{t_{ads}}$ **(s)**	17.3	16.05	97.81	64.48
$\boldsymbol{Q_{feed}}$	$7.44e^{-3}$	$4.52e^{-3}$	$1.08e^{-3}$	$1.06e^{-3}$
$\boldsymbol{L_{module}(m)}$	0.8	0.81	0.81	0.81
$\boldsymbol{D_{module}(m)}$	0.375	0.286	0.15	0.173
$\boldsymbol{n_{feed}}$ **(mol/s)**	1.653	0.385	0.0679	0.0654
Productivity	0.0595	0.0174	0.0135	$6.93e^{-3}$
Specific energy consumption	862	782	550	536
Purity	0.95	0.95	0.95	0.96
Recovery	0.90	0.90	0.90	0.97

As expected, these two types of modules show opposing characteristics. Module type 1 has product productivity 2.5 – 4.4 times higher than that of module type 2. This results in a higher molar flow n_{feed} per module; thus, less number of modules are required to capture CO_2. However, the module has a high energy requirement since the module needs more energy to reach high P_{ads}. Module type 2 has a much lower energy

requirement; however, its productivity is low, which results in a reduced feed flowrate per module. The economic analysis of these modules is presented in section 6.3.

6.2.2 Process Optimization for Natural Gas Power Plants

Natural gas power plants emit flue gas with much diluted CO_2 than that of coal power plants, which makes the capture more energy-intensive. Since our previous work only focused on designing a VPSA system for coal power plants, a new module design is determined for dry flue gas with 4% CO_2 and 96% N_2. The optimization is performed by choosing the specific energy consumption as the objective. Table 23 shows the optimal module design for natural gas power plants. Both module types failed to satisfy all constraints. In particular, the specific energy consumption is very high (>2000 kWh/ton CO_2 captured) to be practical, since the target energy for the post-combustion from a natural gas power plant is ~350 kWh/ton CO_2[175].

Additionally, to see whether we can find any feasible design, we tried relaxing the recovery constraint. DOE guideline requires the purity of the product stream to be at least 95% for downstream post-processing[176]; no requirement exists for product recovery. Thus, we relaxed the recovery constraint to 0.8 and 0.85. However, the energy requirement of a module was still too high and no feasible adsorbent was found. This implies that the use of investigated modules for natural gas power plants is impractical due to their high energy requirement. A study with a different set of adsorbents and a more suitable adsorption cycle (e.g., temperature vacuum swing adsorption) is therefore suggested.

Table 23. Final module design for natural gas power plants

	Zn-MOF-74	13x
P_{ads} (atm)	15.60	19.99
P_{evac} (atm)	0.011	0.012
ω_{ads}	0.41	0.45
t_{ads} (s)	86.96	98.11
Q_{feed} (m^3/s)	$4.47e^{-3}$	$1.03e^{-3}$
L_{module} (m)	0.80	0.80
D_{module} (m)	0.26	0.18
Productivity	$3.43e^{-3}$	$2.19e^{-3}$
Specific energy consumption	2575	2741
Purity	0.85	0.90
Recovery	0.92	0.95

6.3 Economic Analysis of a Modular Facility

Previously, we have successfully obtained 4 promising module designs for coal power plants. To perform economic analysis, we first estimate the cost of a single module. Both the capital and operating costs are estimated to calculate the net present value of a modular facility.

6.3.1 Capital Cost Estimation

The total capital cost of a single module is estimated based on the cost of raw materials and the cost of construction. The raw material cost has two main components: 1) the cost of fiber, and 2) the cost of carbon steel for module casing.

6.3.1.1 Fiber Manufacturing Cost Estimation

The final fiber consists of three main components: adsorbent, polymer, and PCM. The weight fraction of adsorbent in the fiber (ω_{ads}) is determined from optimization. The weight fraction of polymer in the fiber ($\omega_{polymer}$) is fixed at 0.25 to maintain the structural

integrity of the fiber. Table 24 shows the final fiber composition used to estimate the cost of fiber sorbents.

Table 24. Final fiber composition and fiber weight obtained for a single module

	Module Type 1 (max productivity)		Module Type 2 (min energy consumption)	
	Zn-MOF-74	**13x**	**Zn-MOF-74**	**13x**
ω_{ads}	0.15	0.20	0.17	0.25
$\omega_{polymer}$	0.25	0.25	0.25	0.25
ω_{PCM}	0.6	0.55	0.58	0.50
Total fiber weight (kg)	23.81	14.18	3.86	5.25

Fiber manufacturing also requires additional solvents and non-solvents. The required fiber doping composition is obtained from [142]. The amounts of adsorbent, polymer (CA and PVP), and PCM are adjusted accordingly based on ω_{ads}, $\omega_{polymer}$, and ω_{PCM}. The amounts of solvent and non-solvent are calculated based on the total weight of fiber per module (Table 25).

Table 25. Amount of material needed for fiber manufacturing

	Module Type 1 (max productivity)		Module Type 2 (min energy consumption)	
Amount (kg)	**Zn-MOF-74**	**13x**	**Zn-MOF-74**	**13x**
CA polymer	3.32	1.98	0.53	0.73
PVP polymer	2.63	1.56	0.43	0.58
NMP	39.78	23.69	6.45	8.77
H2O	5.23	3.11	0.85	1.15
Adsorbent	3.57	2.85	0.65	1.31
PCM	14.29	7.78	2.25	2.63

Raw material costs are estimated by one of the two methods. If the industrial bulk price is available for any given year, the price is converted to the current year (2020) using cost indices[177]. On the other hand, if the industrial bulk price is not available, the raw

material cost is estimated by converting lab price to industrial bulk price by using the following equation[178]:

$$P_b = P_l \left(\frac{Q_b}{Q_l}\right)^{-0.75} \tag{23}$$

where P_b is the bulk price of 60lb lots in \$/kg, P_l is the lab price in \$/kg, Q_b is the representative bulk amount (60lb or 27216g), and Q_l is the laboratory scale quantity purchased in grams. Table 26 shows the raw material cost required for fiber manufacturing.

Table 26. Raw material cost for fiber manufacturing

Material	Cost (\$/kg)
CA polymer[178]	7.80
PVP polymer	0.37
NMP[178]	13.92
Zeolite 13x[178]	8.86
Zn-MOF-74[179]	146.68
PCM[180]	10
H2O[178]	0.0012

Using Table 25 and Table 26, the final fiber cost per module is obtained and shown in Table 27.

Table 27. Final raw material cost of fiber manufacturing per module

Module Type 1 (max productivity)		Module Type 2 (min energy consumption)	
Zn-MOF-74	**13x**	**Zn-MOF-74**	**13x**
\$1247	\$449	\$211	\$166

6.3.1.2 Module Sizing

The amount of carbon steel needed for module construction is estimated to determine the final cost of a module. From optimization, we have determined the required

module length (L_{module}), module diameter (D_{module}), and the maximum operating pressure (P_{ads}). For a pressure vessel, the required wall thickness is determined from either the hoop stress or the longitudinal stress[177]:

$$t_{hoop} = \frac{PD}{2SE - 1.2P}, t_{longitudinal} = \frac{PD}{4SE + 0.8P} \quad (24)$$

where P is the pressure, D is the diameter, S is the maximum allowable stress, and E is the welded joint efficiency. The final thickness is the maximum of t_{hoop} and $t_{longitudinal}$. S is assumed to be 12900; E is assumed to be 0.85 for a double-welded butt joint. An additional 2mm margin is allowed for wall thickness to account for the corrosion of a vessel over time. Table 28 shows the required wall thickness and the final amount of carbon steel needed to construct a single module.

Table 28. Required wall thickness and amount of carbon steel needed for module construction

	Module Type 1 (max productivity)		**Module Type 2 (min energy consumption)**	
Amount (kg)	**Zn-MOF-74**	**13x**	**Zn-MOF-74**	**13x**
Thickness (mm)	7.11	4.94	2.54	2.86
Carbon steel (kg)	53.56	28.67	7.70	10.02

6.3.1.3 Total capital cost estimation

The total capital cost is estimated using the bare module method to account for all direct and indirect costs for module construction and deployment[177]. A bare module factor of 3.05 is used and multiplied to the sum of the cost of raw materials. The total installation cost of a module is shown in Table 29.

Table 29. Total capital cost per module (module factor = 3.05)

	Module Type 1 (max productivity)		Module Type 2 (min energy consumption)	
Per module	**Zn-MOF-74**	**13x**	**Zn-MOF-74**	**13x**
Total cost	$3960	$1452	$667	$535

6.3.2 Operating Cost Estimation

6.3.2.1 Electricity

Electricity is one major component of the operating cost. The cost of electricity can be estimated by the specific energy consumption shown in **Table 23**. For each module, the total annual energy consumption is calculated as follows:

$$\begin{aligned} &Energy\ consumption\ (kWh) \\ &= Specific\ energy\ consumption\left(\frac{kWh}{ton\ CO_2}\right) \\ &\times CO_{2,captured}(ton\ CO_2) \end{aligned} \tag{25}$$

The cost of electricity is assumed as $0.04/kWh.

6.3.2.2 Maintenance & Labor

The maintenance cost is estimated to be 3.5% of the capital cost[177]. Assuming a modular facility requires 2 operators per shift, the annual cost of direct wages and benefits is calculated by[177]:

$$DW\&B\left(\frac{\$}{yr}\right) = \left(2\frac{operators}{shift}\right) \times (5\ shifts) \times \left(2080\frac{hr}{yr\ operator}\right) \times \left(\frac{\$35}{hr}\right) \quad (26)$$

Therefore, DW&B is estimated to be $\$728{,}000/yr$.

6.3.2.3 Miscellaneous Expense

VPSA module contains fiber sorbents, which need to be replaced as the performance degrades over time. We assume that the fiber sorbents need to be replaced every 24 months. To calculate the cost of replacing fiber sorbents, we assume that it can be estimated by the cost of raw materials needed to construct the fiber. Other incurring costs are assumed to be lumped into maintenance & labor cost.

6.4 Feasibility Study on a Modular Facility: Capture Cost and Energy Penalty

We present a feasibility study on the proposed modular carbon capture facility. Plant Bowen, one of the largest coal power plants in the U.S., is introduced as a representative case study (Table 30).

Table 30. Plant Bowen summary

Capacity (MW)	Generation (GWh)	CO_2 emission (ton/year)
3499	13,583	12,421,585

6.4.1 Net Present Value and Cost of Capture

An economic analysis is performed to determine the cost of CO_2 capture. The total capital cost depends on the number of modules required, which can be calculated as follows:

$$N_{module} = \frac{N_{feed}}{n_{feed}} \tag{27}$$

where N_{feed} is the molar flow rate of the flue gas emitted from the power plant and n_{feed} is the molar flow rate per module.

The net present value (NPV) and equivalent annualized cost (EAC) are calculated assuming a plant life of 20 years and a discount rate of 10%. Table 31 shows the total capital and operating expenses, NPV, and EAC. The cost of capture is obtained by normalizing the annualized cost by the amount of CO_2 captured.

Table 31. Summary of economic analysis on power plant Bowen

	Module Type 1 (max productivity)		**Module Type 2 (min energy consumption)**	
Per module	**Zn-MOF-74**	**13x**	**Zn-MOF-74**	**13x**
\$/module	\$3960	\$1452	\$667	\$535
N_{module}	37,907	162,903	922,455	956,883
Total module cost	\$150M	\$237M	\$616M	\$512M
Electricity cost	\$385M	\$350M	\$246M	\$256M
Maintenance & Labor	\$3.77M	\$6.37M	\$17.74M	\$14.63M
Adsorbent replacement	\$47.29M	\$73.13M	\$195M	\$159M
NPV	\$(3.63B)	\$(3.53B)	\$(3.55B)	\$(3.38B)
EAC	\$(427M)	\$(415M)	\$(417M)	\$(397M)
EAC/ton CO_2	\$(38.15)	\$(37.05)	\$(37.34)	\$(33.24)

The cost of capture per ton of CO_2 seems promising (~\$35/ton CO_2). However, plant Bowen emits a large quantity of flue gas, which means many modules are required

to process all flue gas, especially if the plant uses module type 2 (i.e., 900,000+ modules). Deploying VPSA modules to all power plants in Georgia will require approximately 70 million modules. Thus, it could be difficult to manufacture and install such a large number of modules. A more comprehensive study is needed to evaluate the feasibility of a modular facility.

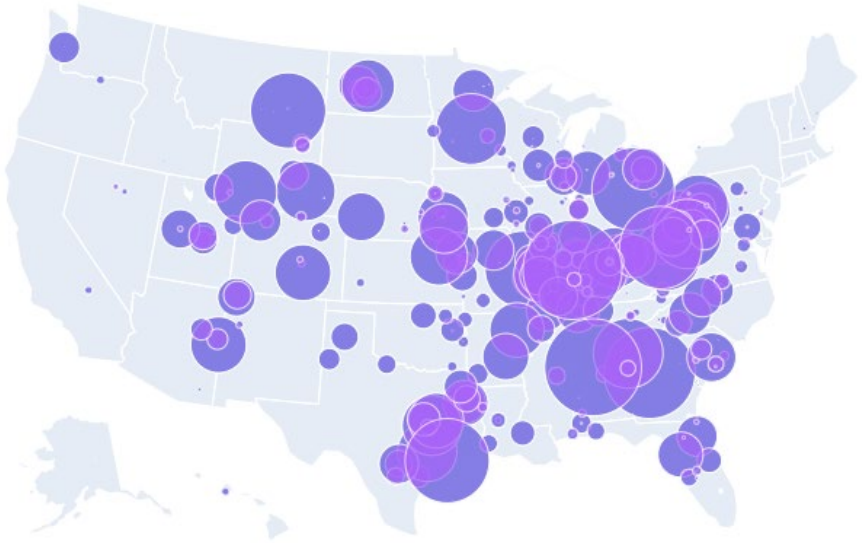


Figure 35. The number of modules required for coal power plants in the U.S. The circle size is proportional to the required number of type 2 module with zeolite 13x.

6.4.2 Energy Penalty

Many carbon capture technologies are currently limited by high energy penalty[140]. Therefore, an important factor that determines the feasibility of a modular facility is the energy requirement. In this work, the energy penalty is defined as the total amount of electricity needed for CCS divided by the total electricity generated by the power plant[175,181]. Table 32 shows the energy penalty for four different modules. The energy penalty of module type 1 is 71% and 64%, respectively; that of module type is 45% and

47%, respectively. While module type 2 has a lower energy penalty, these numbers are still too high for the modules to be implementable.

Table 32. Energy penalty of VPSA modules for plant Bowen

Module Type 1 (max productivity)		Module Type 2 (min energy consumption)	
Zn-MOF-74	**13x**	**Zn-MOF-74**	**13x**
71%	64%	45%	47%

6.5 Conclusion & Future Perspectives

In this chapter, we present a comprehensive study on a VPSA modular facility for carbon capture. We expand chapter 4 by: 1) enhancing the performance of a VPSA module by incorporating additional design variables, 2) exploring power plants with different sources, 3) estimating the capital and operating cost of a modular facility, and 4) performing an economic analysis for a feasibility study. A case study on plant Bowen is presented. While the normalized cost of capture is promising (~\$35/ton of CO_2 captured), a modular facility is still hindered by the high energy penalty. Therefore, we can conclude that deploying these modules to actual power plants is not promising at the moment and more improvements are needed. This study only provides a preliminary cost estimate of a modular facility; thus, a more comprehensive cost analysis is suggested for future work. Furthermore, the cost of capture can be further reduced by exploring different options for selling the captured CO_2 (e.g., enhanced oil recovery).

CHAPTER 7. CONCLUSION

7.1 Conclusion

Surrogate-based optimization has recently gained wide attention in the chemical industry, which is linked to the development in the fields of machine learning, digitization, and data storage systems. Unlike equation-based optimization, the surrogate-based approach only relies on process data generated from a high-fidelity simulation. This data is used to construct a low-fidelity surrogate, or machine learning, model, which is much easier to optimize.

This thesis focuses on studying various strategies for surrogate-based optimization, ranging from data sampling to model construction and optimization. In chapter 3, we explored various SSR techniques to construct low-complexity surrogate models. We compared the performance of SSR models with that of GP and concluded that while SSR models perform well for low-dimensional problems, GP is superior for high-dimensional and high-complexity problems. In chapter 4, we proposed a novel algorithmic framework for surrogate-based MINLPs, which does not require relaxing the integrality constraint. This algorithm also allows parallel searching for multiple promising binary solutions, can incorporate known/explicit constraints for gray-box problems, and supports various types of machine learning models and optimization solvers. We also propose the best sampling strategy for MINLP problems. The algorithm has been released as an open-source Python library.

In the final two chapters, we present a case study on modular carbon capture. In chapter 5, we perform simultaneous adsorbent selection and process optimization using the surrogate-based MINLP algorithm. The algorithm successfully identified the best adsorbent among 75 possible options and accurately determined the optimal operating conditions. In chapter 6, we present an extensive analysis of a modular facility. The capital and operating cost of a single module is calculated for several promising module designs. An economic analysis is performed to determine the feasibility of a modular facility. We conclude that the proposed modular facility has an energy requirement that is still too high to be practical.

7.2 Future Work

For future work, two main areas can be explored. One area is algorithm improvement for surrogate-based optimization. The algorithm has been so far tested on mid-size MINLP problems (up to 29 dimension) and can be expanded to higher-dimensional problems. This requires: 1) a more efficient algorithmic framework to find a solution within a reasonable time, 2) a more flexible and accurate surrogate model that can approximate complex physical systems, and 3) extensive testing of the algorithm on a set of high-dimensional benchmark problems. These enhancements will allow one to use the proposed algorithm for high-dimensional problems, which are often seen in real-life problems.

The second area is modular process intensification for carbon capture. Several challenges currently remain to commercially deploy the modules to power plants; hence, three main improvements are suggested. First, the efficiency of a module should be improved to lower the energy penalty. Currently, the energy requirement of a VPSA

module is too high to be implementable. Therefore, lowering the energy requirement is a critical step. Second, several assumptions are made in this work to obtain a preliminary cost of a modular facility. A more extensive cost analysis is therefore suggested. Finally, a supply chain network optimization study is proposed. After we obtain a more efficient module design, a supply chain network optimization can allow one to determine how and what type of module should be deployed to minimize the cost of capture. The supply chain network can be further enhanced by including the transportation and storage cost and exploring various options for the usage of captured CO_2. This study will provide valuable insights into modular carbon capture.

APPENDIX

A. Process Synthesis Case Study: Original Problem Formulation adapted from [131]

$$minimize\ z = 5y_1 + 8y_2 + 6y_3 + 10y_4 + 6y_5 + 7y_6 + 4y_7 + 5y_8 - 10x_3 - 15x_5 + 15x_{10} + 80x_{!7} + 25x_{19} + 35x_{21} - 40x_9 + 15x_{14} - 35x_{25} + \exp(x_3) + \exp\left(\frac{x_5}{1.2}\right) - 6.5\ln(x_{10} + x_{17} + 1)$$

$$s.t. -1.5\ln(x_{19} + 1) - \ln(x_{21} + 1) - x_{14} \leq 0$$

$$-\ln(x_{10} + x_{17} + 1) \leq 0$$

$$-x_3 - x_5 + x_{10} + 2x_{17} + 0.8x_{19} + 0.8x_{24} - 0.5x_9 - x_{14} - 2x_{25} \leq 0$$

$$-x_3 - x_5 + 2x_{17} + 0.8x_{19} + 0.8x_{21} - 2x_9 - x_{14} - 2x_{25} \leq 0$$

$$-2x_{17} - 0.8x_{19} - 0.8x_{21} + 2x_9 + x_{14} + 2x_{25} \leq 0$$

$$-0.8x_{19} - 0.8x_{21} + x_{14} \leq 0$$

$$-x_{17} + x_9 + x_{25} \leq 0$$

$$-0.4x_{14} - 0.4x_{21} + 1.5x_{14} \leq 0$$

$$0.16x_{19} + 0.16x_{21} - 1.2x_{14} \leq 0$$

$$x_{10} - 0.8x_{17} \leq 0$$

$$-x_{10} + 0.4x_{17} \leq 0$$

$$\exp(x_3) - 10y_1 \leq 1$$

$$\exp\left(\frac{x_5}{1.2}\right) - 10y_2 \leq 1$$

$$x_9 - 10y_3 \leq 0\ (G)$$

$$0.8x_{19} + 0.8x_{21} - 10y_4 \leq 0$$

$$2x_{17} - 2x_9 - 2x_{25} - 10y_5 \leq 0$$

$$x_{19} - 10y_6 \leq 0\ (G)$$

$$x_{21} - 10y_7 \leq 0\ (G)$$

$$x_{10} + x_{17} - 10y_8 \leq 0\ (G)$$

$$y_1 + y_2 = 1\ (G), y_4 + y_5 \leq 1(G)$$

$$-y_4 + y_6 + y_7 = 0\ (G), y_3 - y_8 \leq 0\ (G)$$

$$y \in \{0,1\}^8, a \leq x \leq b, x = \left(x_j : j = 3,5,10,17,19,21,9,14,25\right) \in \mathbb{R}^9$$

$$a^T = \{0,0,0,0,0,0,0,0,0\}, b^T = \{2,2,1,2,2,2,2,1,3\}$$

B. Process Synthesis Case Study: Gray-box Formulation in the scaled space

$$minimize\ \hat{f}\left(x_i^{'}, d_j\right)$$

$$s.t.\ \hat{g}_c\left(x_i^{'}, d_j\right) \leq 0$$

$$2x_9^{'} - 10\left(\frac{1 - d_{6,0}}{d_{6,1} - d_{6,0}}\right) \leq 0$$

$$2x_{19}^{'} - 10\left(\frac{1 - d_{7,0}}{d_{7,1} - d_{7,0}}\right) \leq 0$$

$$x_{10}^{'} + 2x_{17}^{'} - 10\left(\frac{1 - d_{8,0}}{d_{8,1} - d_{8,0}}\right) \leq 0$$

$$\left(\frac{1 - d_{1,0}}{d_{1,1} - d_{1,0}}\right) + \left(\frac{1 - d_{2,0}}{d_{2,1} - d_{2,0}}\right) = 1$$

$$\left(\frac{1 - d_{4,0}}{d_{4,1} - d_{4,0}}\right) + \left(\frac{1 - d_{5,0}}{d_{5,1} - d_{5,0}}\right) \leq 1$$

$$-\left(\frac{1-d_{4,0}}{d_{4,1}-d_{4,0}}\right)+\left(\frac{1-d_{6,0}}{d_{6,1}-d_{6,0}}\right)+\left(\frac{1-d_{7,0}}{d_{7,1}-d_{7,0}}\right)\leq 1$$

$$\left(\frac{1-d_{3,0}}{d_{3,1}-d_{3,0}}\right)-\left(\frac{1-d_{8,0}}{d_{8,1}-d_{8,0}}\right)\leq 0$$

$$d_{j,0}+d_{j,1}=1, j=1,\dots,8$$

$$d_j\in\{0,1\}^{16}, 0\leq x_i^{'}\leq 1$$

REFERENCES

1. Boukouvala F, Floudas CA. ARGONAUT: AlgoRithms for Global Optimization of coNstrAined grey-box compUTational problems. *Optimization Letters.* 2017;11(5):895-913.
2. Cozad A, Sahinidis NV, Miller DC. Learning surrogate models for simulation-based optimization. *AIChE Journal.* 2014;60(6):2211-2227.
3. Amaran S, Sahinidis NV, Sharda B, Bury SJ. Simulation optimization: a review of algorithms and applications. *4OR.* 2014;12(4):301-333.
4. Tekin E, Sabuncuoglu I. Simulation optimization: A comprehensive review on theory and applications. *IIE Transactions.* 2004;36(11):1067-1081.
5. Bhosekar A, Ierapetritou M. Advances in surrogate based modeling, feasibility analysis, and optimization: A review. *Computers & Chemical Engineering.* 2018;108:250-267.
6. Bajaj I, Iyer SS, Faruque Hasan MM. A trust region-based two phase algorithm for constrained black-box and grey-box optimization with infeasible initial point. *Computers & Chemical Engineering.* 2017.
7. Forrester AIJ, Keane AJ. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences.* 2009;45(1):50-79.
8. Amaran S, Sahinidis NV, Sharda B, Bury SJ. Simulation optimization: a review of algorithms and applications. *Annals of Operations Research.* 2016;240(1):351-380.
9. Boukouvala F, Misener R, Floudas CA. Global optimization advances in Mixed-Integer Nonlinear Programming, MINLP, and Constrained Derivative-Free Optimization, CDFO. *European Journal of Operational Research.* 2016;252(3):701-727.
10. McBride K, Sundmacher K. Overview of Surrogate Modeling in Chemical Process Engineering. *Chemie Ingenieur Technik.* 2019;91(3):228-239.
11. Rios LM, Sahinidis NV. Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization.* 2013;56(3):1247-1293.
12. Hüllen G, Zhai J, Kim SH, Sinha A, Realff MJ, Boukouvala F. Managing Uncertainty in Data-Driven Simulation-Based Optimization. *Computers & Chemical Engineering.* 2019:106519.
13. Boukouvala F, Hasan MMF, Floudas CA. Global optimization of general constrained grey-box models: new method and its application to constrained PDEs for pressure swing adsorption. *Journal of Global Optimization.* 2017;67(1):3-42.
14. Cozad A, Sahinidis NV, Miller DC. A combined first-principles and data-driven approach to model building. *Computers & Chemical Engineering.* 2015;73:116-127.

15. Davis E, Ierapetritou M. A kriging-based approach to MINLP containing black-box models and noise. *Industrial & Engineering Chemistry Research.* 2008;47(16):6101-6125.
16. Davis SE, Cremaschi S, Eden MR. Efficient Surrogate Model Development: Impact of Sample Size and Underlying Model Dimensions. In: Eden MR, Ierapetritou MG, Towler GP, eds. *Computer Aided Chemical Engineering.* Vol 44. Elsevier; 2018:979-984.
17. Garud SS, Karimi IA, Kraft M. LEAPS2: Learning based Evolutionary Assistive Paradigm for Surrogate Selection. *Computers & Chemical Engineering.* 2018;119:352-370.
18. Graciano JEA, Le Roux GAC. Improvements in surrogate models for process synthesis. Application to water network system design. *Computers & Chemical Engineering.* 2013;59:197-210.
19. Keßler T, Kunde C, McBride K, et al. Global optimization of distillation columns using explicit and implicit surrogate models. *Chemical Engineering Science.* 2019;197:235-245.
20. Kim SH, Boukouvala F. Machine learning-based surrogate modeling for data-driven optimization: a comparison of subset selection for regression techniques. *Optimization Letters.* 2019.
21. Schweidtmann AM, Mitsos A. Deterministic Global Optimization with Artificial Neural Networks Embedded. *Journal of Optimization Theory and Applications.* 2018.
22. Dias LS, Ierapetritou MG. Integration of planning, scheduling and control problems using data-driven feasibility analysis and surrogate models. *Computers & Chemical Engineering.* 2020;134:106714.
23. Mencarelli L, Pagot A, Duchêne P. Surrogate-based modeling techniques with application to catalytic reforming and isomerization processes. *Computers & Chemical Engineering.* 2020;135:106772.
24. Mencarelli L, Chen Q, Pagot A, Grossmann IE. A review on superstructure optimization approaches in process system engineering. *Computers & Chemical Engineering.* 2020;136:106808.
25. Schweidtmann AM, Huster WR, Lüthje JT, Mitsos A. Deterministic global process optimization: Accurate (single-species) properties via artificial neural networks. *Computers & Chemical Engineering.* 2019;121:67-74.
26. Beykal B, Avraamidou S, Pistikopoulos IPE, Onel M, Pistikopoulos EN. DOMINO: Data-driven Optimization of bi-level Mixed-Integer NOnlinear Problems. *Journal of Global Optimization.* 2020.
27. Garud SS, Karimi IA, Kraft M. Smart Sampling Algorithm for Surrogate Model Development. *Computers & Chemical Engineering.* 2017;96:103-114.
28. Wilson ZT, Sahinidis NV. Automated learning of chemical reaction networks. *Computers & Chemical Engineering.* 2019;127:88-98.
29. Zantye MS, Arora A, Faruque Hasan MM. Operational power plant scheduling with flexible carbon capture: A multistage stochastic optimization approach. *Computers & Chemical Engineering.* 2019;130:106544.
30. Tso WW, Demirhan CD, Floudas CA, Pistikopoulos EN. Multi-scale energy systems engineering for optimal natural gas utilization. *Catalysis Today.* 2019.

31. Balasubramanian P, Bajaj I, Hasan MMF. Simulation and optimization of reforming reactors for carbon dioxide utilization using both rigorous and reduced models. *Journal of CO2 Utilization.* 2018;23:80-104.
32. Jakobsson S, Patriksson M, Rudholm J, Wojciechowski A. A method for simulation based optimization using radial basis functions. *Optimization and Engineering.* 2010;11(4):501-532.
33. Boukouvala F, Muzzio FJ, Ierapetritou MG. Dynamic Data-Driven Modeling of Pharmaceutical Processes. *Industrial & Engineering Chemistry Research.* 2011;50(11):6743-6754.
34. Bittante A, Pettersson F, Saxén H. Optimization of a small-scale LNG supply chain. *Energy.* 2018;148:79-89.
35. Sampat AM, Martin E, Martin M, Zavala VM. Optimization formulations for multi-product supply chain networks. *Computers & Chemical Engineering.* 2017;104:296-310.
36. Beykal B, Boukouvala F, Floudas CA, Sorek N, Zalavadia H, Gildin E. Global Optimization of Grey-Box Computational Systems Using Surrogate Functions and Application to Highly Constrained Oil-Field Operations. *Computers & Chemical Engineering.*
37. Ciaurri DE, Mukerji T, Durlofsky LJ. Derivative-Free Optimization for Oil Field Operations. In: Yang X-S, Koziel S, eds. *Computational Optimization and Applications in Engineering and Industry.* Berlin, Heidelberg: Springer Berlin Heidelberg; 2011:19-55.
38. Jansen JD, Durlofsky LJ. Use of reduced-order models in well control optimization. *Optimization and Engineering.* 2017;18(1):105-132.
39. Isebor OJ, Durlofsky LJ, Echeverría Ciaurri D. A derivative-free methodology with local and global search for the constrained joint optimization of well locations and controls. *Computational Geosciences.* 2014;18(3):463-482.
40. Khoury GA, Smadbeck J, Kieslich CA, et al. Princeton_TIGRESS 2.0: High refinement consistency and net gains through support vector machines and molecular dynamics in double-blind predictions during the CASP11 experiment. *Proteins: Structure, Function, and Bioinformatics.* 2017;85(6):1078-1098.
41. Liwo A, Lee J, Ripoll DR, Pillardy J, Scheraga HA. Protein structure prediction by global optimization of a potential energy function. *Proceedings of the National Academy of Sciences.* 1999;96(10):5482.
42. DiMaio F, Terwilliger TC, Read RJ, et al. Improved molecular replacement by density- and energy-guided protein structure optimization. *Nature.* 2011;473:540.
43. Hastie T, Tibshirani R, Friedman JH. *The Elements of Statistical Learning.* 2 ed. Springer2009.
44. Booker AJ, Dennis JE, Frank PD, Serafini DB, Torczon V, Trosset MW. A rigorous framework for optimization of expensive functions by surrogates. *Structural optimization.* 1999;17(1):1-13.
45. Yondo R, Andrés E, Valero E. A review on design of experiments and surrogate models in aircraft real-time and many-query aerodynamic analyses. *Progress in Aerospace Sciences.* 2018;96:23-61.

46. Zhang K-S, Han Z-H, Gao Z-J, Wang Y. Constraint aggregation for large number of constraints in wing surrogate-based optimization. *Structural and Multidisciplinary Optimization.* 2019;59(2):421-438.
47. Nelder JA, Mead RA. *A Simplex Method for Function Minimization Comput.* Vol 71965.
48. Torczon V. On the Convergence of Pattern Search Algorithms. *SIAM Journal on Optimization.* 1997;7(1):1-25.
49. Audet C, Dennis J. Mesh Adaptive Direct Search Algorithms for Constrained Optimization. *SIAM Journal on Optimization.* 2006;17(1):188-217.
50. Jones DR, Perttunen CD, Stuckman BE. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications.* 1993;79(1):157-181.
51. Huyer W, Neumaier A. Global Optimization by Multilevel Coordinate Search. *Journal of Global Optimization.* 1999;14(4):331-355.
52. Audet C, Dennis J. Pattern Search Algorithms for Mixed Variable Programming. *SIAM Journal on Optimization.* 2001;11(3):573-594.
53. Abramson MA, Audet C, Chrissis JW, Walston JG. Mesh adaptive direct search algorithms for mixed variable optimization. *Optimization Letters.* 2008;3(1):35.
54. Liuzzi G, Lucidi S, Rinaldi F. Derivative-Free Methods for Mixed-Integer Constrained Optimization Problems. *Journal of Optimization Theory and Applications.* 2015;164(3):933-965.
55. Müller J, Shoemaker C, Piche R. SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems. *Computers and Operations Research.* 2013;40(5):1383-1400.
56. Powell MJD. UOBYQA: unconstrained optimization by quadratic approximation. *Mathematical Programming.* 2002;92(3):555-582.
57. Gilmore P, Kelley C. An Implicit Filtering Algorithm for Optimization of Functions with Many Local Minima. *SIAM Journal on Optimization.* 1995;5(2):269-285.
58. Conn A, Scheinberg K, Vicente L. *Introduction to Derivative-Free Optimization.* Society for Industrial and Applied Mathematics; 2009.
59. Rashid K, Ambani S, Cetinkaya E. An adaptive multiquadric radial basis function method for expensive black-box mixed-integer nonlinear constrained optimization. *Engineering Optimization.* 2013;45(2):185-206.
60. Holmström K, Quttineh N-H, Edvall MM. An adaptive radial basis algorithm (ARBF) for expensive black-box mixed-integer constrained global optimization. *Optimization and Engineering.* 2008;9(4):311-339.
61. Davis E, Ierapetritou M. A kriging based method for the solution of mixed-integer nonlinear programs containing black-box functions. *Journal of Global Optimization.* 2009;43(2):191-205.
62. Müller J. MISO: mixed-integer surrogate optimization framework. *Optimization and Engineering.* 2016;17(1):177-203.
63. Ben-Tal A, Zibulevsky M. Penalty/Barrier Multiplier Methods for Convex Programming Problems. *SIAM Journal on Optimization.* 1997;7(2):347-366.
64. Deb K. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering.* 2000;186(2):311-338.

65. Jones DR, Schonlau M, Welch WJ. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization.* 1998;13(4):455-492.
66. Regis RG, Shoemaker CA. Constrained Global Optimization of Expensive Black Box Functions Using Radial Basis Functions. *Journal of Global Optimization.* 2005;31(1):153-171.
67. Heaton J. *Introduction to Neural Networks for JAVA.* 2 ed: Heaton Research, Inc.; 2008.
68. !!! INVALID CITATION !!! .
69. Wang C, Duan Q, Gong W, Ye A, Di Z, Miao C. An evaluation of adaptive surrogate modeling based optimization with two benchmark problems. *Environmental Modelling & Software.* 2014;60:167-179.
70. Fen C-S, Chan C, Cheng H-C. Assessing a Response Surface-Based Optimization Approach for Soil Vapor Extraction System Design. *Journal of Water Resources Planning and Management.* 2009;135(3):198-207.
71. Jones DR. A Taxonomy of Global Optimization Methods Based on Response Surfaces. *Journal of Global Optimization.* 2001;21(4):345-383.
72. Palmer K, Realff M. Metamodeling Approach to Optimization of Steady-State Flowsheet Simulations: Model Generation. *Chemical Engineering Research and Design.* 2002;80(7):760-772.
73. Anand P, Siva Prasad BVN, Venkateswarlu CH. MODELING AND OPTIMIZATION OF A PHARMACEUTICAL FORMULATION SYSTEM USING RADIAL BASIS FUNCTION NETWORK. *International Journal of Neural Systems.* 2009;19(02):127-136.
74. Jeong S, Murayama M, Yamamoto K. *Efficient Optimization Design Method Using Kriging Model.* Vol 422005.
75. Miller AJ. Selection of Subsets of Regression Variables. *Journal of the Royal Statistical Society Series A (General).* 1984;147(3):389-425.
76. Candès EJ, Romberg JK, Tao T. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics.* 2006;59(8):1207-1223.
77. Guyon I, Weston J, Barnhill S, Vapnik V. Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning.* 2002;46(1):389-422.
78. Feng G, Guo J, Jing B-Y, Sun T. Feature subset selection using naive Bayes for text classification. *Pattern Recognition Letters.* 2015;65:109-115.
79. Wright J, Yang AY, Ganesh A, Sastry SS, Ma Y. Robust Face Recognition via Sparse Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2009;31(2):210-227.
80. Sahinidis N. The ALAMO approach to machine learning. In: Kravanja Z, Bogataj M, eds. *Computer Aided Chemical Engineering.* Vol 38. Elsevier; 2016:2410.
81. Cozad A, Sahinidis N, Miller D. *A combined first-principles and data-driven approach to model building.* Vol 732015.
82. Gorissen D, Couckuyt I, Demeester P, Dhaene T, Crombecq K. A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design. *J Mach Learn Res.* 2010;11:2051-2055.

83. Tawarmalani M, Sahinidis NV. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming.* 2005;103(2):225-249.
84. Hastie T, Tibshirani R, Wainwright M. *Statistical Learning with Sparsity.* New York: Chapman and Hall/CRC; 2015.
85. Ren H, vs. L1 Convex Optimization in Sparse Coding: Comparative Study in Abnormal Event Detection G. *Greedy vs. L1 Convex Optimization in Sparse Coding: Comparative Study in Abnormal Event Detection.* 2015.
86. Tibshirani R. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society Series B (Methodological).* 1996;58(1):267-288.
87. Zou H, Hastie T. Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society Series B (Statistical Methodology).* 2005;67(2):301-320.
88. Hastie T, Qian J. Glmnet Vignette. https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html. Published 2014. Accessed2018.
89. Zou H, Hastie T, Tibshirani R. Sparse Principal Component Analysis. *Journal of Computational and Graphical Statistics.* 2006;15(2):265-286.
90. Kawano S, Fujisawa H, Takada T, Shiroishi T. Sparse principal component regression with adaptive loading. *Computational Statistics & Data Analysis.* 2015;89:192-203.
91. Geladi P, Kowalski BR. Partial least-squares regression: a tutorial. *Analytica Chimica Acta.* 1986;185:1-17.
92. Chun H, Keleş S. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society Series B, Statistical Methodology.* 2010;72(1):3-25.
93. Guyon I, Elisseeff A. An introduction to variable and feature selection. *J Mach Learn Res.* 2003;3:1157-1182.
94. Smola AJ, Schölkopf B. A tutorial on support vector regression. *Statistics and Computing.* 2004;14(3):199-222.
95. Cherkassky V, Ma Y. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks.* 2004;17(1):113-126.
96. Friedman JH, Hastie T, Tibshirani R, Simon N, Narasimhan B, Qian J. Package 'glmnet': Lasso and Elastic-Net Regularized Generalized Linear Models. https://cran.r-project.org/web/packages/glmnet/glmnet.pdf. Published 2018. Accessed.
97. Zou H. Package 'elasticnet': Elastic-Net for Sparse Estimation and Sparse PCA. https://cran.r-project.org/web/packages/elasticnet/elasticnet.pdf. Published 2015. Accessed.
98. Kawano S. Package 'spcr': Sparse Principal Component Regression. https://cran.r-project.org/web/packages/spcr/spcr.pdf. Published 2016. Accessed.
99. Chung D, Chun H, Keles S. An Introduction to the 'spls' Package, Version 1.0. 2012.
100. Chung D, Chun H, Keleş S. An Introduction to the ‘spls’ Package, Version 1.0. https://cran.r-project.org/web/packages/spls/vignettes/spls-example.pdf. Published 2018. Accessed.

101. Karatzoglou A, Smola AJ, Hornik K. Package 'kernlab': Kernel-Based Machine Learning Lab. https://cran.r-project.org/web/packages/kernlab/kernlab.pdf. Published 2018. Accessed.
102. Kuhn M. Package 'caret': Classification and Regression Training. https://cran.r-project.org/web/packages/caret/caret.pdf. Published 2018. Accessed.
103. Drud A. CONOPT. GAMS. https://www.gams.com/latest/docs/S_CONOPT.html. Accessed2018.
104. Kim SH, Boukouvala F. Surrogate-based optimization for mixed-integer nonlinear problems. *Computers & Chemical Engineering.* 2020;140:106847.
105. Zhai J, Boukouvala F. Nonlinear Variable Selection Algorithms for Surrogate Modeling. *AIChE Journal.* 2019;0(ja):e16601.
106. Caballero JA, Grossmann IE. An algorithm for the use of surrogate models in modular flowsheet optimization. *AIChE Journal.* 2008;54(10):2633-2650.
107. Henao CA, Maravelias CT. Surrogate-based superstructure optimization framework. *AIChE Journal.* 2011;57(5):1216-1232.
108. Sangbum L, En Sup Y, Grossmann IE. Superstructure optimization of chemical process. Paper presented at: SICE 2003 Annual Conference (IEEE Cat. No.03TH8734); 4-6 Aug. 2003, 2003.
109. Larson J, Leyffer S, Palkar P, Wild SM. A Method for Convex Black-Box Integer Global Optimization. *arXiv preprint arXiv:190311366.* 2019.
110. Deep K, Singh KP, Kansal ML, Mohan C. A real coded genetic algorithm for solving integer and mixed integer optimization problems. *Applied Mathematics and Computation.* 2009;212(2):505-518.
111. Liuzzi G, Lucidi S, Rinaldi F. Derivative-free methods for bound constrained mixed-integer optimization. *Computational Optimization and Applications.* 2012;53(2):505-526.
112. McKay MD, Beckman RJ, Conover WJ. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics.* 1979;21(2):239-245.
113. Owen AB. ORTHOGONAL ARRAYS FOR COMPUTER EXPERIMENTS, INTEGRATION AND VISUALIZATION. *Statistica Sinica.* 1992;2(2):439-452.
114. Sobol IM. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics.* 1967;7(4):86-112.
115. Swiler LP, Hough PD, Qian P, Xu X, Storlie C, Lee H. Surrogate Models for Mixed Discrete-Continuous Variables. In: Ceberio M, Kreinovich V, eds. *Constraint Programming and Decision Making.* Cham: Springer International Publishing; 2014:181-202.
116. Cocchi G, Pillo G, Fasano G, et al. DFL - A Derivative-Free Library. http://www.iasi.cnr.it/~liuzzi/DFL/index.php/news-list. Published 2019. Accessed 05/10/19.
117. Reeves C. *Genetic Algorithms for the Operations Researcher.* Vol 91997.
118. Eberhart R, Kennedy J. A new optimizer using particle swarm theory. Paper presented at: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science; 4-6 Oct. 1995, 1995.

119. Romeo F, Sangiovanni-Vincentelli A. A theoretical framework for simulated annealing. *Algorithmica.* 1991;6(1):302.
120. Egea JA, Henriques D, Cokelaer T, et al. MEIGO: an open-source software suite based on metaheuristics for global optimization in systems biology and bioinformatics. *BMC Bioinformatics.* 2014;15(1):136.
121. Mistry M, Letsios D, Krennrich G, Lee RM, Misener R. Mixed-Integer Convex Nonlinear Optimization with Gradient-Boosted Trees Embedded. *arXiv e-prints.* 2018. https://ui.adsabs.harvard.edu/abs/2018arXiv180300952M. Accessed March 01, 2018.
122. Qian PZG, Wu H, Wu CFJ. Gaussian Process Models for Computer Experiments With Qualitative and Quantitative Factors. *Technometrics.* 2008;50(3):383-396.
123. Gramacy RB, Lee HKH. Bayesian Treed Gaussian Process Models With an Application to Computer Modeling. *Journal of the American Statistical Association.* 2008;103(483):1119-1130.
124. Eason J, Cremaschi S. Adaptive sequential sampling for surrogate model generation with artificial neural networks. *Computers & Chemical Engineering.* 2014;68:220-232.
125. Brownlee J. Why One-Hot Encode Data in Machine Learning? https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/. Published 2017. Accessed.
126. McCaffrey J. Neural Network Data Normalization and Encoding. https://visualstudiomagazine.com/articles/2013/07/01/neural-network-data-normalization-and-encoding.aspx. Published 2013. Accessed.
127. Rall D, Menne D, Schweidtmann AM, et al. Rational design of ion separation membranes. *Journal of Membrane Science.* 2019;569:209-219.
128. Grossmann I, Viswanathan J, Vecchietti A, Raman R, Kalvelagen E. *GAMS/DICOPT: A discrete continuous optimization package.* Vol 112002.
129. Drud AS. CONOPT—A Large-Scale GRG Code. *ORSA Journal on Computing.* 1994;6(2):207-216.
130. MINLPLib: A library of mixed-integer and continuous nonlinear programming instances. 2019. http://www.minlplib.org/. Accessed May 7, 2019.
131. Duran MA, Grossmann IE. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming.* 1986;36(3):307-339.
132. Choi S, Drese JH, Jones CW. Adsorbent Materials for Carbon Dioxide Capture from Large Anthropogenic Point Sources. *ChemSusChem.* 2009;2(9):796-854.
133. Hasan MMF, First EL, Boukouvala F, Floudas CA. A multi-scale framework for CO2 capture, utilization, and sequestration: CCUS and CCU. *Computers & Chemical Engineering.* 2015;81:2-21.
134. Bhown AS, Freeman BC. Analysis and Status of Post-Combustion Carbon Dioxide Capture Technologies. *Environmental Science & Technology.* 2011;45(20):8624-8632.
135. Ben-Mansour R, Habib MA, Bamidele OE, et al. Carbon capture by physical adsorption: Materials, experimental investigations and numerical modeling and simulations – A review. *Applied Energy.* 2016;161:225-255.

136. Hasan MMF, Boukouvala F, First EL, Floudas CA. Nationwide, Regional, and Statewide CO2 Capture, Utilization, and Sequestration Supply Chain Network Optimization. *Industrial & Engineering Chemistry Research.* 2014;53(18):7489-7506.
137. Bui M, Adjiman CS, Bardow A, et al. Carbon capture and storage (CCS): the way forward. *Energy & Environmental Science.* 2018;11(5):1062-1176.
138. U.S. EPA's Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990-2017. EPA. https://www.epa.gov/ghgemissions/inventory-us-greenhouse-gas-emissions-and-sinks. Accessed2019.
139. Yang H, Xu Z, Fan M, et al. Progress in carbon dioxide separation and capture: A review. *Journal of Environmental Sciences.* 2008;20(1):14-27.
140. Hasan MMF, First EL, Floudas CA. Cost-effective CO2 capture based on in silico screening of zeolites and process optimization. *Physical Chemistry Chemical Physics.* 2013;15(40):17601-17618.
141. Leperi KT, Chung YG, You F, Snurr RQ. Development of a General Evaluation Metric for Rapid Screening of Adsorbent Materials for Postcombustion CO2 Capture. *ACS Sustainable Chemistry & Engineering.* 2019;7(13):11529-11539.
142. DeWitt SJA, Rubiera Landa HO, Kawajiri Y, Realff M, Lively RP. Development of Phase-Change-Based Thermally Modulated Fiber Sorbents. *Industrial & Engineering Chemistry Research.* 2019;58(15):5768-5776.
143. Ruthven DM. *Principles of Adsorption and Adsorption Processes.* Wiley; 1984.
144. Ebner AD, Ritter JA. State-of-the-art adsorption and membrane separation processes for carbon dioxide production from carbon dioxide emitting industries. 2009.
145. Rezaei F, Subramanian S, Kalyanaraman J, Lively RP, Kawajiri Y, Realff MJ. Modeling of rapid temperature swing adsorption using hollow fiber sorbents. *Chemical Engineering Science.* 2014;113:62-76.
146. Jang H-Y, Johnson JR, Ma Y, Mathias R, Bhandari DA, Lively RP. Torlon® hollow fiber membranes for organic solvent reverse osmosis separation of complex aromatic hydrocarbon mixtures. *AIChE Journal.* 2019;65(12):e16757.
147. Sinha A, Realff MJ. A parametric study of the techno-economics of direct CO2 air capture systems using solid adsorbents. *AIChE Journal.* 2019;65(7):e16607.
148. Samanta A, Zhao A, Shimizu GKH, Sarkar P, Gupta R. Post-Combustion CO2 Capture Using Solid Sorbents: A Review. *Industrial & Engineering Chemistry Research.* 2012;51(4):1438-1463.
149. Darunte LA, Sen T, Bhawanani C, et al. Moving Beyond Adsorption Capacity in Design of Adsorbents for CO2 Capture from Ultradilute Feeds: Kinetics of CO2 Adsorption in Materials with Stepped Isotherms. *Industrial & Engineering Chemistry Research.* 2019;58(1):366-377.
150. Findley JM, Ravikovitch PI, Sholl DS. The Effect of Aluminum Short-Range Ordering on Carbon Dioxide Adsorption in Zeolites. *The Journal of Physical Chemistry C.* 2018;122(23):12332-12340.
151. Huck JM, Lin L-C, Berger AH, et al. Evaluating different classes of porous materials for carbon capture. *Energy & Environmental Science.* 2014;7(12):4132-4146.

152. Yazaydın AÖ, Snurr RQ, Park T-H, et al. Screening of Metal−Organic Frameworks for Carbon Dioxide Capture from Flue Gas Using a Combined Experimental and Modeling Approach. *Journal of the American Chemical Society.* 2009;131(51):18198-18199.
153. Subramanian Balashankar V, Rajendran A. Process Optimization-Based Screening of Zeolites for Post-Combustion CO2 Capture by Vacuum Swing Adsorption. *ACS Sustainable Chemistry & Engineering.* 2019;7(21):17747-17755.
154. Khurana M, Farooq S. Adsorbent Screening for Postcombustion CO2 Capture: A Method Relating Equilibrium Isotherm Characteristics to an Optimum Vacuum Swing Adsorption Process Performance. *Industrial & Engineering Chemistry Research.* 2016;55(8):2447-2460.
155. Khurana M, Farooq S. Integrated adsorbent-process optimization for carbon capture and concentration using vacuum swing adsorption cycles. *AIChE Journal.* 2017;63(7):2987-2995.
156. Agarwal A, Biegler LT, Zitney SE. A superstructure-based optimal synthesis of PSA cycles for post-combustion CO2 capture. *AIChE Journal.* 2010;56(7):1813-1828.
157. Kikkinides ES, Yang RT, Cho SH. Concentration and recovery of carbon dioxide from flue gas by pressure swing adsorption. *Industrial & Engineering Chemistry Research.* 1993;32(11):2714-2720.
158. Ko D, Siriwardane R, Biegler LT. Optimization of Pressure Swing Adsorption and Fractionated Vacuum Pressure Swing Adsorption Processes for CO2 Capture. *Industrial & Engineering Chemistry Research.* 2005;44(21):8084-8094.
159. Haghpanah R, Majumder A, Nilam R, et al. Multiobjective Optimization of a Four-Step Adsorption Process for Postcombustion CO2 Capture Via Finite Volume Simulation. *Industrial & Engineering Chemistry Research.* 2013;52(11):4249-4265.
160. Fiandaca G, Fraga E, Brandani S. A multi-objective genetic algorithm for the design of pressure swing adsorption. *Fiandaca, G and Fraga, ES and Brandani, S (2009) A multi-objective genetic algorithm for the design of pressure swing adsorption Engineering Optimization, 41 (9) pp 833-854 ISSN 0305215X.* 2009;41.
161. Haghpanah R, Nilam R, Rajendran A, Farooq S, Karimi IA. Cycle synthesis and optimization of a VSA process for postcombustion CO2 capture. *AIChE Journal.* 2013;59(12):4735-4748.
162. Baldea M, Edgar Thomas F, Stanley Bill L, Kiss Anton A. Modular manufacturing processes: Status, challenges, and opportunities. *AIChE Journal.* 2017;63(10):4262-4272.
163. Kim Y-h, Park LK, Yiacoumi S, Tsouris C. Modular Chemical Process Intensification: A Review. *Annual Review of Chemical and Biomolecular Engineering.* 2017;8(1):359-380.
164. Stankiewicz A, Moulijn JA. *Process Intensification: Transforming Chemical Engineering.* Vol 962000.

165. Rubiera Landa HO, Lively RP, Kawajiri Y, Realff M. Theoretical investigation of vacuum pressure swing adsorption process applying thermally-modulated fiber composite adsorbents. In:2020.
166. Xiao P, Zhang J, Webley P, Li G, Singh R, Todd R. Capture of CO2 from flue gas streams with zeolite 13X by vacuum-pressure swing adsorption. *Adsorption.* 2008;14(4):575-582.
167. Schiesser WE. *The Numerical Method of Lines: Integration of Partial Differential Equations.* Academic Press; 1991.
168. Gear CW, Gear WC. *Numerical Initial Value Problems in Ordinary Differential Equations.* Prentice-Hall; 1971.
169. MATLAB. Natick, Massachusetts: The MathWorks Inc.; 2018.
170. Rangaiah GP, Bonilla-Petriciolet A. *Multi-Objective Optimization in Chemical Engineering: Developments and Applications.* Wiley; 2013.
171. Alin A. Multicollinearity. *WIREs Computational Statistics.* 2010;2(3):370-374.
172. Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer New York; 2013.
173. Electricity in the United States. https://www.eia.gov/energyexplained/electricity/electricity-in-the-us.php. Published 2020. Accessed November 3, 2020.
174. Electricity. In. U.S. Energy Information Administration2020.
175. Vasudevan S, Farooq S, Karimi IA, Saeys M, Quah MCG, Agrawal R. Energy penalty estimates for CO2 capture: Comparison between fuel types and capture-combustion modes. *Energy.* 2016;103:709-714.
176. James R, Zoelle A, Keairns D, Turner M, Woods M, Kuehn N. Cost and Performance Baseline for Fossil Energy Plants. In: Laboratory NET, ed. Vol Volume 1: Bituminous Coal and Natural Gas to Electricity2019.
177. Seider WD, Seader JD, Lewin DR, Widagdo S. *Product and Process Design Principles: Synthesis, Analysis, and Evaluation.* Wiley; 2010.
178. Chen G, Lively RP, Jones CW, Koros WJ. Fiber Adsorbents for Odorant Removal from Pipeline Grade Natural Gas. *Industrial & Engineering Chemistry Research.* 2014;53(17):7113-7120.
179. MOF-74-Zn Supplemental Information. Millipore Sigma. https://www.sigmaaldrich.com/technical-documents/articles/materials-science/metal-organic-frameworks/mof-74-zn.html. Accessed November 4, 2020.
180. Stephen J. A. DeWitt RA, Hector Rubiera-Landa, Jongwoo Park, Yoshiaki Kawajiri†, David S. Sholl, Matthew Realff, Ryan P Lively. Analysis of Energetics and Economics of Sub-ambient Hybrid Post-Combustion CO2 Capture. In:2020.
181. House KZ, Harvey CF, Aziz MJ, Schrag DP. The energy penalty of post-combustion CO2 capture & storage and its implications for retrofitting the U.S. installed base. *Energy & Environmental Science.* 2009;2(2):193-205.