Preprint Cambridge Centre for Computational Chemical Engineering ISSN 1473 – 4273

Machine learning approach for constructing surrogates of a biodiesel plant flow sheet model

Xueheng Qiu [1], Janusz J. Sikorski [2], Sushant S. Garud [3], Ponnuthurai Nagaratnam Suganthan [1], Markus Kraft [2,4]

released: 9 June 2017

[1] School of Electrical
and Electronic Engineering
Nanyang Technological University
50 Nanyang Avenue
Singapore 639798
Singapore
Email: epnsugan@e.ntu.edu.sg

[2] Department of Chemical Engineering
and Biotechnology
University of Cambridge
New Museums Site, Pembroke Street
Cambridge, CB2 3RA
United Kingdom
E-mail: mk306@cam.ac.uk

[3] Department of Chemical and
Biomolecular Engineering
National University of Singapore
4 Engineering Drive 4
Singapore, 117576
Singapore

[4] School of Chemical and
Biomedical Engineering
Nanyang Technological University
62 Nanyang Drive
Singapore, 637459
Singapore
E-mail: mk306@cam.ac.uk

Preprint No. 184

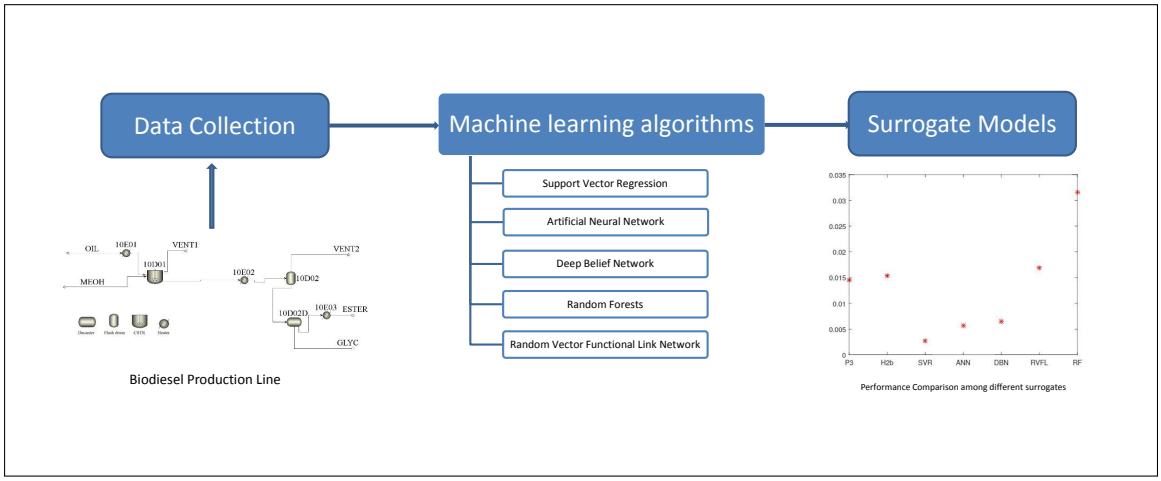
Keywords: Process flow sheet model, Surrogate models, Biodiesel, Support vector regression, Neural networks, Random forests

Edited by
Computational Modelling Group
Department of Chemical Engineering and Biotechnology
University of Cambridge
New Museums Site
Pembroke Street
Cambridge CB2 3RA
United Kingdom

Fax: + 44 (0)1223 334796
E-Mail: c4e@cam.ac.uk
World Wide Web: http://como.ceb.cam.ac.uk/

Abstract

In this work, a process flowsheet of biodiesel plant is used to construct surrogate models by various machine learning methods. For this paradigm, we approximate the heat duties of various units in the flowsheet by varying $1, 2, 6$ and 11 input variables. We use 3 domain sizes of input variables, and 5 different surrogates namely support vector regression (SVR), artificial neural network (ANN), deep belief network (DBN), random forests (RF), and random vector functional link network (RVFL). A comparison among polynomial response surface fitting, high dimensional model representation (HDMR) and machine learning models is performed based on three error measures namely root-mean-squared-deviation (RMSD), R^2 and residuals. Moreover, the efficiency of learning methods is compared based on their computational expense and some statistical measure. The simulation results show the attractiveness of machine learning methods for constructing surrogate models in context of process simulation.



Highlights:

- Input-output relations within process flowsheet of a biodiesel plant are analyzed.
- Various machine learning algorithms are employed for constructing surrogate models.
- A variety of scenarios are considered including four input dimensions and five surrogates.
- Polynomial response surface fitting and high dimensional model representation are compared with machine learning methods.
- The simulation results show the effectiveness and efficiency of machine learning methods.

Contents

1 Introduction

Mathematical models play an important role in the design and operation of chemical plants. Two often conflicting aims must be met. The plant needs to maximize its profits and at the same time minimize its negative impact on the environment. Typical environmental constraints include the reduction of wastes and pollutants and reducing the carbon foot print. Operating plants in vicinity to each other have led to the concept of the eco-industrial park in which synergistic effects such as sharing waste heat can be exploited. As a result, eco-industrial parks (EIPs) have attracted public interests and become popular with the development of the concepts of "industrial ecology" [17, 29], "industry symbiosis" [12, 38], and "sustainable development" [8].

In recent years, numerous research works concerning various aspects of EIPs have been published, which mainly focus on optimal design of sustainable industrial activities by constructing mathematical models to create exchange networks of resources among the members of EIPs [13, 31, 32, 36]. However, it is very challenging and expensive to perform the holistic modelling of complex and highly interconnected networks such as EIPs, that include many physical models with disparate processes. This problem may be tackled by the concept of "Industry 4.0" as discussed by Pan *et al.* [38], wherein they suggest that the physical models can be replaced by their surrogate models to reduce the computational expense and memory significantly. Moreover, this would make dynamic modelling and optimization possible. Figure 1 is the framework of EIP modelling based on Industry 4.0.

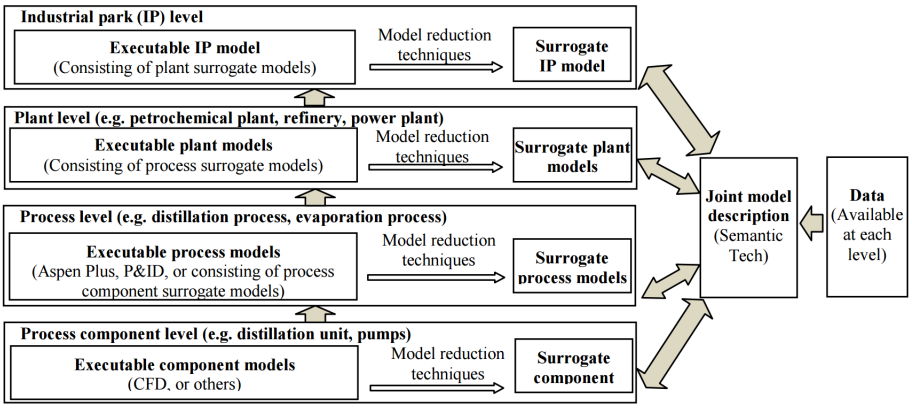


Figure 1: *Framework of EIP modelling based on Industry* 4.0 *(Adopted from [38]).*

A surrogate model is a reasonably accurate and computationally cheap approximation of expensive high fidelity experiment and/or computer simulation [20]. Surrogate models are typically used to obtain the behaviour of the input-output relationships, deal with the noise or missing data, and assist in design and optimization of computer experiments. The literature is replete with the applications of surrogate models in various fields of engineering and science like modelling [11], sensitivity analysis [3], space exploration [21], parameter estimation [33], optimization in circuit design, nanoparticle synthesis, flood monitoring [43] *etc*. Model selection is the most important part for constructing a surrogate model and there are numerous modelling techniques in the literature, which include

polynomial fitting [44], kriging [34], artificial neural network (ANN) [15], support vector machine (SVM) [20] and so on. A detailed review on data sampling, surrogate model generation techniques and their performance measures can be found in [45].

Sikorski *et al.*[44] employ two surrogate models *viz.* polynomial response surface and (HDMR) for parametrisation of typical process flowsheet of a biodiesel plant. Moreover, they investigate the effects of dimensionality, domain size, and surrogate type on the accuracy of surrogate models in a variety of scenarios. In this paper, we construct accurate surrogate models using machine learning algorithms, instead of polynomial and HDMR fitting, using 11 inputs and 6 outputs in a typical biodiesel plant. The same simulation scenarios as in [44] are considered: $1, 2, 6$ and 11 input variables and 3 domain sizes. Overall, 5 different machine learning techniques (support vector regression (SVR), artificial neural network (ANN), deep belief network (DBN), random forests (RF) and random vector functional link network (RVFL)) are used for constructing surrogates. These machine learning algorithms are compared based on both accuracy and efficiency.

This paper is organized as follows. Section 2 explains the theoretical background on machine learning methods followed by section 3 that discusses the details of experiment setup, including biodiesel plant simulation, data collection, implementation and performance estimation. The experimental results and comparison are discussed in Section 4. Finally, section 5 presents our conclusions and future works.

2 Background

2.1 Support Vector Regression

Support vector machine (SVM) is a machine learning algorithm proposed by Cortes and Vapnik [14] based on statistical learning theory. Structural risk minimization is the basic concept of this method. A version of SVM for regression is proposed in [19], which is called SVR.

Consider a training data set given in Eq. 1 for constructing a surrogate model.

$$D = \left\{(x^{(k)}, y^{(k)})\right\}, 1, 2, \cdots, l \tag{1}$$

where $x^{(k)} \in \mathbb{R}^n$ is the k^{th} input vector with n elements, $y^{(k)} \in \mathbb{R}$ is the corresponding output data and l is the number of training samples. The regression function can be defined as

$$y'(x) = w^T \phi(x) + b \tag{2}$$

where w is the weight vector, b is the bias, and $\phi(x)$ maps the input vector x to a higher dimensional feature space. w and b can be obtained by solving the following optimization problem:

$$\min_{w,b,\xi,\xi^*} \frac{1}{2} w^T w + C \sum_{k=1}^{l} (\xi_k + \xi_k^*) \tag{3}$$

Subject to:

$$\begin{aligned} & y^{(k)} - w^T \phi(x^{(k)}) - b \leq \epsilon + \xi_k \\ & w^T \phi(x^{(k)}) + b - y^{(k)} \leq \epsilon + \xi_k^* \\ & \xi_k, \xi_k^* \geq 0 \end{aligned} \tag{4}$$

where C is a predefined positive trade-off parameter between model simplicity and generalization ability, ξ_k and ξ_k^* are the slack variables measuring the cost of the errors.

For nonlinear input data set, kernel functions can be used to map from original space onto a higher dimensional feature space in which a linear regression model can be built. The dual optimization problem is

$$\min_{\alpha,\alpha^*} \frac{1}{2}(\alpha - \alpha^*)^T Q(\alpha - \alpha^*) + \epsilon \sum_{k=1}^{l}(\alpha_k + \alpha_k^*) + \sum_{k=1}^{l} y^{(k)}(\alpha_k - \alpha_k^*) \tag{5}$$

Subject to:

$$\begin{aligned} & e^T(\alpha - \alpha^*) = 0, \\ & 0 \leq \alpha_k, \alpha_k^* \leq C \end{aligned} \tag{6}$$

where $e = [1, \ldots, 1]^T$ is the vector of all ones, Q is an l by l positive semidefinite matrix, K is the kernel function, $Q_{ij} = K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$.

Thus, the final SVR function is obtained as follows

$$y'(x) = \sum_{k=1}^{l}(\alpha_k^* - \alpha_k) K(x^{(k)}, x) + b \tag{7}$$

where α_k and α_k^* are the Lagrange multipliers. The most frequently used kernel function is the Gaussian radial function (RBF) with a width of σ

$$K(x_i, x_j) = \exp(-\left\|x_i - x_j\right\|^2 / (2\sigma^2)) \tag{8}$$

In this study, we use ϵ-SVR provided by LIBSVM [9], and choose RBF kernel function with parameters chosen by a grid search. As suggested by the authors of LIBSVM toolbox, exponentially growing sequences of the prediefined positive trade-off parameter C and the tolerance of termination criterion ϵ are used for parameter selection. The range of C is $[2^{-4}, 2^4]$, and the range of ϵ is $[10^{-3}$,$10^{-1}]$. All the other parameters remain at default settings in LIBSVM.

2.2 Artificial Neural Network

ANN is a learning model inspired by human brain, especially the central nervous system [22]. The simplest version of ANN is called single-hidden layer feedforward neural network (SLFN). Figure 2 is an illustration of a three-layer SLFN. There are three fundamental layers in a SLFN: an input layer with the same number of neurons as the dimension of input features; a hidden layer comprised of neurons with nonlinear activation function;

and an output layer which aggregates the outputs from the hidden layer neurons. The output from SLFN is:

$$y'(x) = g(\sum_{j=1}^{h} w_{jo} v_j + b_{ho}), \quad j = 1, \ldots, m \tag{9}$$

$$v_j = f(\sum_{i=1}^{n} w_{ij} x_i + b_{ih}), \quad i = 1, \ldots, n \tag{10}$$

where x_i is the input to the neuron i_i; $f()$ and $g()$ are nonlinear activation functions; v_j is the output of hidden layer neuron h_j; y is the output of this SLFN; n and m are the number of input features and the number of the hidden layer neurons respectively; w_{ij} is the weight of the connection between the input variable i_i and the neuron h_j of the hidden layer; w_{jo} is the weight of the connection between the hidden layer neuron h_j and the output; b_{ih} and b_{ho} are the biases.

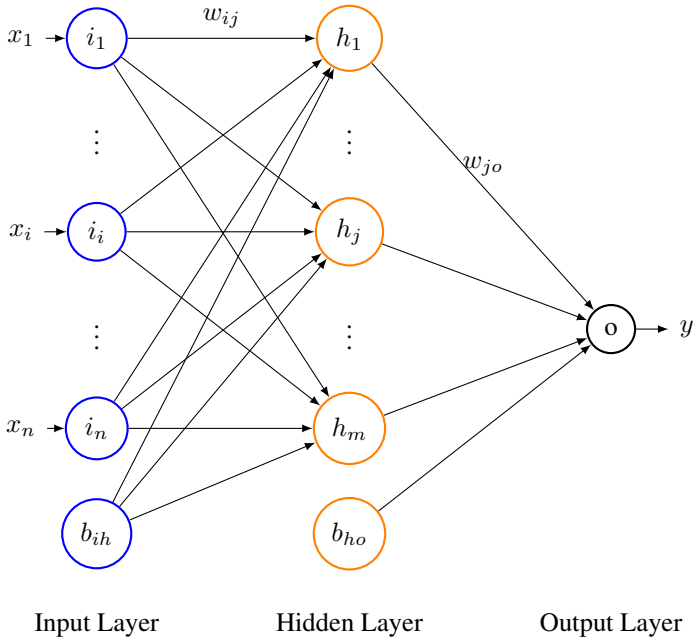


Figure 2: *Schematic of a neural network model*

While training an SLFN, random values are assigned to the weights and then the weights are tuned by back-propagation (BP) [35] or using a closed form solution [42, 51].

In this work, a deep learning toolbox in Matlab is used for constructing neural networks [37]. The size of neural networks is determined by the size of input dimension. For input layer, the number of input neurons $n = 11$ for 11-dimensional surrogates, while the number of hidden size neurons is twice of the input layer neurons, *e.g.* $m = 22$ for 11-dimensional surrogates. The activation function of hidden layer $f()$ is optimal $\tanh$, while the activation function of output layer $g()$ is sigmoid. Moreover, the number of

iterations for back propagation is set to 3000, and we take a mean gradient step over a batch of samples with the size of 10. All the other parameters remain at default settings in the deep learning toolbox.

2.3 Random Vector Functional Link Network

RVFL network is a randomized version of the SLFN, which has direct connections between input and output neurons (functional link). It uses fixed random weights and closed form least square estimation instead of the BP to tune the weights [39, 40]. Figure 3 shows the schematic of an RVFL network.

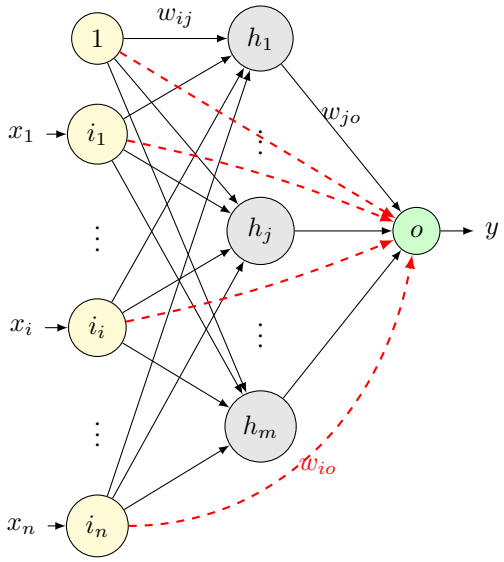


Figure 3: *Schematic of an RVFL network. The dashed arrows show the direct connections between the input neurons and the output neurons, whose weights are denoted as w_{io}.*

It is worth noting that all hidden layer weights w_{ij} in an RVFL are generated with uniformly distributed random values within the interval $[-S,+S]$, where S is a scale factor to be determined during the parameter tuning stage [50]. Therefore, the output v_j from hidden neuron h_j can be calculated based on the activation function. Here, logistic sigmoid function is used as an example:

$$v_j = \text{logsig}(\sum_{i=1}^{n} w_{ij}x_i + b_{ih}), \quad i = 1, \ldots, n \tag{11}$$

where x_i is the training data.

The output layer weight vector w_o, which includes both w_{io} and w_{jo}, need to be determined by certain optimization method. Due to the efficiency of closed-form solution, RVFL employs least square estimation to calculate the output layer weights:

$$w_o = (v^T v)^{-1} v^T y \tag{12}$$

where y is the training target vector. Thus, the predicted values can be calculated by applying the obtained w_o and w_{ij} to testing data [41]:

$$y' = w_o \cdot \text{logsig}(w_{ij} \cdot x) \tag{13}$$

Herein, RVFL is developed by the authors in Matlab based on the work in [50]. According to suggestions in [42, 50], the randomization uses a uniform distribution in $[-1, 1]$, the logistic sigmoid function is used and the number of hidden neurons is selected over $1000 : 10000$ with a step-size of 1000.

2.4 Deep Belief Network (DBN)

Deep learning is a branch of machine learning that attempts to model high-level abstractions in data by using model architectures, with complex structures with multiple non-linear transformations [4]. Deep learning algorithms are fundamentally based on distributed representations, which means that observed data can be represented by interactions of many different factors on different levels. The main promise of deep learning is replacing handcrafted features with efficient algorithms for unsupervised feature extraction [46].

DBN proposed by [23] provides a new way to train deep generative models, which is called layer-wise greedy pre-training algorithm. Figure 4 shows the flowchart of DBN and there is no inter-connection between units in each layer. A restricted Boltzmann machine (RBM) is a neural network which can learn the probability distribution over the input datasets. Figure 5 shows the network structure of an RBM. The DBN pre-training procedure treats each consecutive pair of layers in the MLP as a restricted Boltzmann machine (RBM) [26] whose joint probability distribution is defined as $p(v, h) = \frac{1}{Z} \cdot \exp(-E(v, h))$ with the energy function

$$E(v, h) = -\sum_{i=1}^{p}\sum_{j=1}^{q} w_{ij} h_i v_j - \sum_{j=1}^{q} b_j v_j - \sum_{i=1}^{p} a_i h_i \tag{14}$$

where $Z = \sum_{v,h} \exp(-E(v, h))$, $V = (V_1, \ldots, V_q)$ represent q visible units, $H = (H_1, \ldots, H_p)$ are p hidden units, w_{ij} is a real valued weight associated with the edge between units V_j and H_i, and b_j and a_i are real valued bias terms associated with the j^{th} visible and the i^{th} hidden variable, respectively. In binary RBMs, the random variables (V, H) take values $(v, h) \in \{0, 1\}^{p+q}$ [47].

The conditional probability of a single variable is

$$\begin{aligned} p(H_i = 1|v) &= \sigma(\sum_{j=1}^{q} w_{ij} v_j + a_i) \\ p(V_j = 1|h) &= \sigma(\sum_{i=1}^{p} w_{ij} h_i + b_j) \end{aligned} \tag{15}$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid activation function.

The RBM parameters can be efficiently trained in an unsupervised fashion by maximizing the log-likelihood with the approximate contrastive divergence algorithm [24]

$$\ln \mathcal{L}(\theta|v) = \ln \sum_{h} \exp(-E(v,h)) - \ln \sum_{v,h} \exp(-E(v,h)) \tag{16}$$

where θ is the parameters of the model.

The gradient of the log-likelihood of a single training pattern $\boldsymbol{v}$ are given as:

$$\frac{\partial \ln \mathcal{L}(\theta|v)}{\partial w_{ij}} = p(H_i = 1|v)v_j - \sum_{v} p(v)p(H_i = 1|v)v_j \tag{17}$$

$$\frac{\partial \ln \mathcal{L}(\theta|v)}{\partial a_i} = p(H_i = 1|v) - \sum_{v} p(v)p(H_i = 1|v) \tag{18}$$

$$\frac{\partial \ln \mathcal{L}(\theta|v)}{\partial b_j} = v_j - \sum_{v} p(v)v_j \tag{19}$$

In order to train multiple layers, one trains the first layer, freezes it, and uses the conditional expectation of the output as the input to the next layer and continues training next layers. Hinton and many others have found that initializing MLPs with pretrained parameters never hurts and often helps [23, 25].

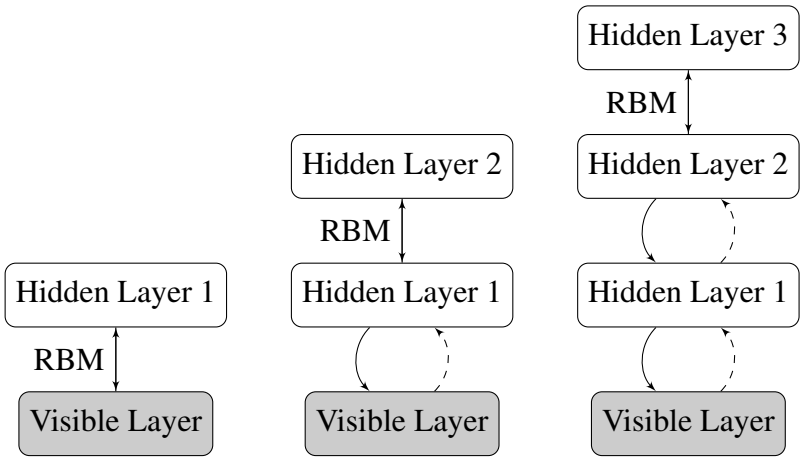


Figure 4: *Flowchart of a Deep Belief Network (DBN)*

In this study, DBN is implemented using the same deep learning toolbox as ANN. Two RBMs are stacked for pre-training with the size of visible and hidden layers as five times of input dimension, *e.g.* $[55\ 55]$ for 11-dimensional surrogates. The learning rate or momentum is set as 0.3. Then an SLFN with the same parameter settings as in Section 2.2 is used to generate final outputs.

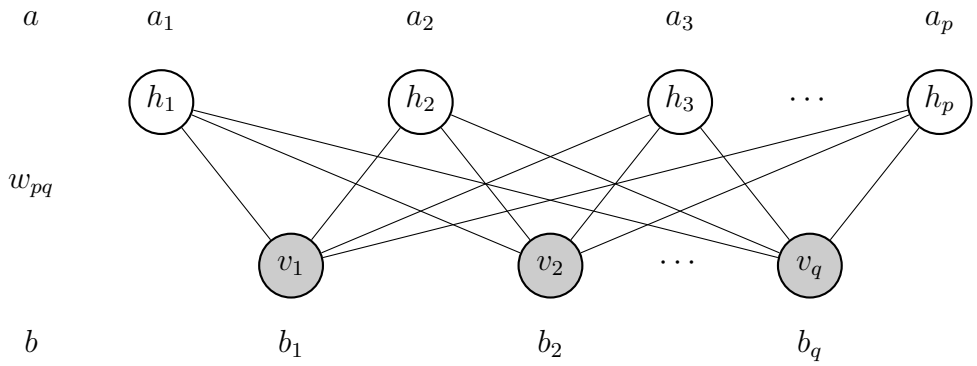


Figure 5: *Schematic Diagram of a Restricted Boltzmann Machine (RBM)*

2.5 Random Forests

Random forests, or random decision forests [27, 28], proposed by Breiman [6], is an ensemble learning method for both of classification and regression problems. Random forests combine bagging and random subspace method (RSM) by conducting random feature subspace at each node of the classification and regression tree (CART) [7]. Bagging (bootstrap aggregating), also developed by Breiman [5], is a widely used ensemble method. In bagging ensemble method, one trains each weak learning machine on bootstrap samples of the original training samples, then aggregating the outputs. RSM is a combining method which trains the learning machines on randomly chosen subspaces of the original input space, and combines the outputs by a majority vote or median [28]. More specifically, at each node of the decision tree in random forest, m features from total n input features are randomly selected. Then according to an impurity criterion, one of these features is used to perform a partition along the feature axis [7]. The algorithm of RF is presented in [48, 49].

In this work, RF is developed using the function "Treebagger" in the classification and regression trees (CART) package in Matlab. We set the parameter "NumPredictorsToSample" as one third of the number of input features to invoke RF algorithm. The number of decision trees L is set as 2000.

3 Experimental setup

3.1 Biodiesel plant simulation

Aspen Plus is a process modelling and optimisation software used by the bulk, fine, specialty, and biochemical industries, as well as the polymer industry for the design, operation, and optimisation of safe, profitable manufacturing facilities [1]. We simulate the biodiesel plant process using Aspen Plus v8.6. The process flowsheet model under investigation considers two steps of the biodiesel production process: a reaction step and a separation step. Figure 6 shows Aspen Plus process flowsheet model. The final fuel *i.e.* fatty acid methyl ester, is produced via trans-esterification where triglycerides react

with methanol to form methyl ester and glycerin in the presence of an alkaline catalyst. The flowsheet is based on an existing plant designed by Lurgi GmbH. It consists of the following units: a continuously stirred tank reactor (CSTR), a flash drum, a decanter, 3 exchangers and 11 material streams. In this process, palm oil is reacted with methanol in CSTR to produce glycerol and methylpalmitate (biodiesel) and then passed through a flash drum and a decanter to separate excess methanol and glycerol. The steady-state simulation provides a wide variety of chemical and physical information ranging from throughput to heat duties of individual unit.

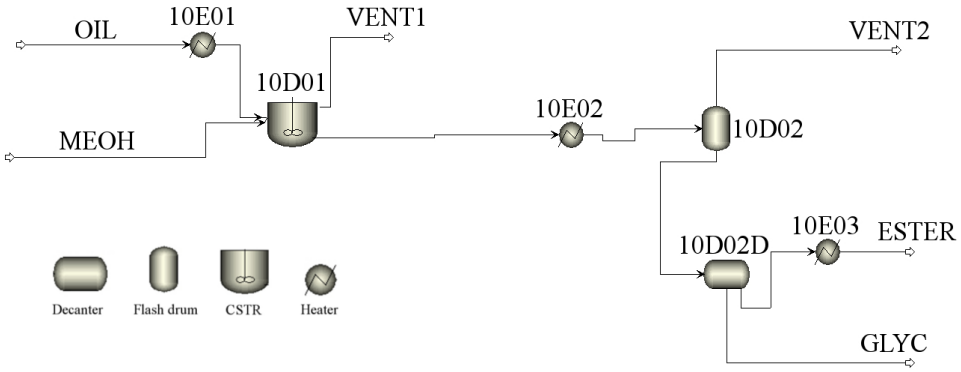


Figure 6: *Graphical representation of the process flow sheet model of a biodiesel production line. Adopted from [44].*

3.2 Data collection

In this study, the Model Development Suite (MoDS) [2], the custom-made Python 3.4, and R 3.2.2 scripts are used for data collection, processing and visualisation. MoDS is an advanced software toolkit designed to analyse black-box models (e.g. executables, batch scripts). Generation of input-output data is the critical process to ensure high accuracy of the surrogate model. Therefore, a sufficient number of sample points and a suitable sampling method are required to describe the input-output relation satisfactorily for a given number of independent variables and operation range. Sobol sequence, a quasi-random low discrepancy sampling method, is employed for sampling. To this end, data samples are generated using the following procedure:

1. Sobol sequence is used to generate input data for user-specified variables.
2. These input data are used to evaluate the simulation and generate corresponding outputs.
3. MoDS retrieves the user-specified outputs.
4. Input-output data is scanned for errors and necessary corrections are carried out.
5. Machine learning algorithms are used to construct surrogate models using the input-output datasets.

The workflow of MoDS is shown in Figure 7. A variety of scenarios are considered: $1, 2, 6$ and 11 input variables are changed simultaneously, 3 different domain sizes of the input variables are considered and 5 different machine learning methods are used. We use 400 points per input variable for evaluating the simulation. These input-output datasets are subsequently used for fitting surrogates and calculating R^2 and $\bar{R}^2$. Additionally, test sets with 100 points per dimension are generated for computing RMSD and residuals. In this study, three domain sizes of the input variables are considered in order to assess their effect on the parametrisation accuracy. The definitions of input and output variables are summarised in Table 1 and Table 2, respectively.

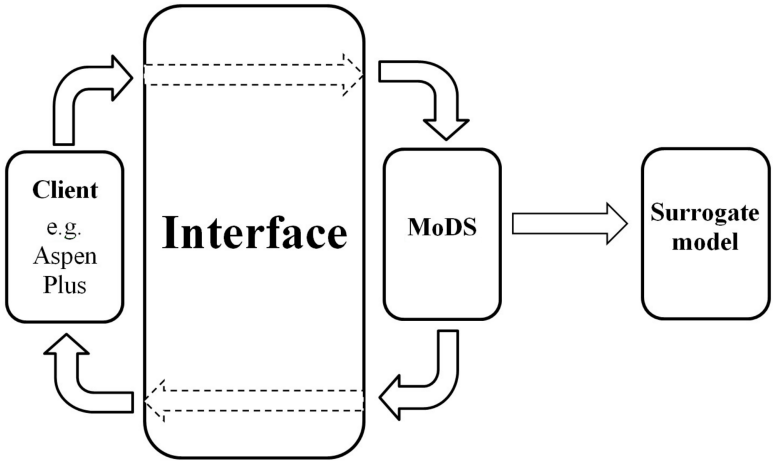


Figure 7: *Model Development Suite work flow. Adopted from [44].*

Table 1: *Definitions and domain bounds of input variables*

Name	Lower bounds	Upper bounds	Operation point
Molar flow of tripalmitine oil (kmol/h)	20, 22.5, 25	40, 37.5, 35	30
Temperature of tripalmitine oil (°C)	20, 22.5, 25	40, 37.5, 35	30
Operation temperature of CSTR 10D01 (°C)	44, 49, 54	64	60
Volume of CSTR 10D01 (m^3)	40, 43, 45	50, 49, 47	45
Operation temperature of flash drum 10D02 (°C)	80, 82.5, 85	100, 97.5, 95	90
Operation temperature of heater 10E01 (°C)	60, 62.5, 65	80, 77.5, 75	70
Molar flow of methanol (kmol/h)	150, 160, 170	210, 200, 190	180
Temperature of methanol (°C)	20, 22.5, 25	40, 37.5, 35	30
Operation temperature of decanter 10D02D (°C)	20, 22.5, 25	40, 37.5, 35	30
Operation temperature of heater 10E02 (°C)	80, 82.5, 85	100, 97.5, 95	90
Operation temperature of heater 10E03 (°C)	60, 62.5, 65	80, 77.5, 75	70

3.3 Data normalization

Before the machine learning algorithms being used to construct the surrogate models, all the training and testing values are linearly scaled to $[0, 1]$. The scaling formula is:

$$\bar{y}^{(i)} = \frac{y_{max} - y^{(i)}}{y_{max} - y_{min}} \tag{20}$$

Table 2: *Definitions of Output variables*

Name
Heat duty of heater 10E01 (MW)
Heat duty of heater 10E02 (MW)
Heat duty of heater 10E03 (MW)
Heat duty of reactor 10D01 (MW)
Heat duty of flash drum 10D02 (MW)
Heat duty of decanter 10D02D (MW)

3.4 Performance estimation

A number of error measures are used to evaluate the performance of surrogate models: R^2, root-mean-squared-deviation (RMSD) and residual plots. They are defined as follows:

$$\begin{aligned} RMSD &= \sqrt{\frac{1}{l_s}\sum_{j=1}^{l_s}(y'^{(j)} - y(j))^2} \\ R^2 &= 1 - \frac{\sum_{i=1}^{l}(y^{(i)} - y'^{(i)})^2}{\sum_{i=1}^{l}(y^{(i)} - \bar{y})^2} \\ e_j &= y_j - y'^{(j)} \end{aligned} \tag{21}$$

where $y'^{(i)}$ and $y'^{(j)}$ are the predicted values of corresponding training data $y^{(i)}$ and testing data $y^{(j)}$, respectively; $\bar{y}$ is the empirical mean of training data points, l is the number of training data samples, l_s is the number of testing data points, $e^{(j)}$ is the residual for j^{th} testing data point, $i = 1, \ldots, l$ and $j = 1, \ldots, l_s$.

RMSD is the sample standard deviation of the differences between predicted values and observed values [30]. It is a good metric for comparing predictive power of different models for a particular variable, but not between the variables due to scale dependency. The coefficient of determination, R^2, is a measure indicating fit of statistical model to the data [18]. Moreover, residual plots are more informative than the other error measures with single number indices.

4 Results and Comparison

4.1 Performance comparison using R^2 values for training data

R^2 is an error measure which compares the discrepancies between the predicted data and actual data with the discrepancies between the arithmetic average and actual data. Figure 8 shows the plots of R^2 values for the surrogates constructed for heat duty of reactor 10D01 with 11 input variables by various machine learning methods. As all the surrogate models perform quite well and achieve R^2 values higher than 0.99, it is very difficult to differentiate between the models by R^2 using training data.

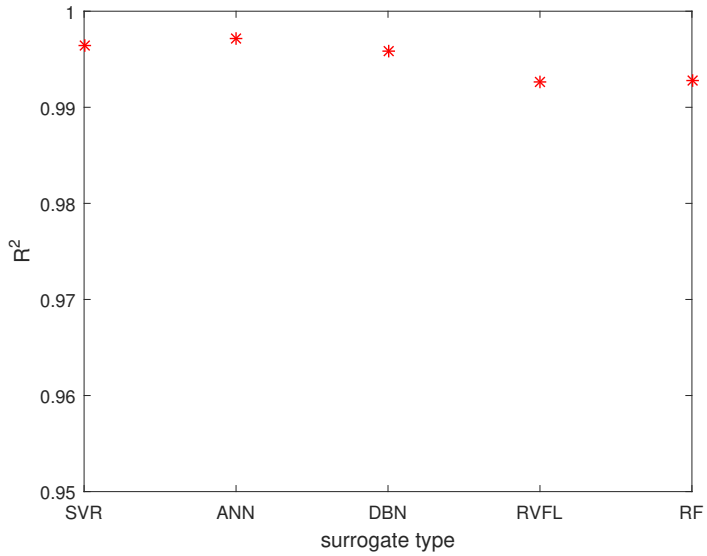


Figure 8: *Plot of R^2 for the surrogate models*

4.2 Performance comparison using RMSD values for testing data

The RMSD values for different machine learning algorithms are listed in Table 3, and the optimal parameters selected for surrogate construction are shown in Table 4. A number of observations and conclusions can be made. Firstly, all the constructed surrogate models achieve at least a reasonable fit regardless of the domain size and number of dimensions, showing the effectiveness of the machine learning methods.

Moreover, statistical tests are employed to analyze and rank the performance of the learning models. The Friedman test ranks the algorithms for each dataset separately, and then assigns average ranks in case of ties. The null-hypothesis states that all the algorithms have the same performance. If the null-hypothesis is rejected, the Nemenyi post-hoc test is applied to compare all the learning models with each other. This tells whether the performances of any two learning models among total k models is significantly different. The comparative result of statistical test based on RMSD is shown in Figure 9. The methods are arranged in descending order of the performance from top to bottom. Note that the models within a vertical line whose length is less than or equal to a critical distance have statistically the similar performance. The critical distance for Nemenyi test is defined as:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{22}$$

where k is the number of algorithms ($k = 5$ in this experiment), N is the number of data sets ($N = 22$ here), and q_α is the critical value based on the studentized range statistic divided by $\sqrt{2}$ [16].

To this end, we make following key inferences. Firstly, SVR-based surrogate model has achieved the best performance, followed by DBN and ANN. DBS and SFLN performs equally in the most cases except in some cases the former performs worse than the latter.

Table 3: *Performance evaluation of surrogate models with RMSD*

Dimension	Output	Surrogate model				
		SVR	ANN	DBN	RVFL	RF
11	10E01	**2.75E-04**	2.10E-03	1.99E-03	5.91E-03	1.69E-02
	10E02	1.43E-02	1.07E-02	**9.54E-03**	3.61E-02	5.27E-02
	10E03	**1.45E-03**	2.26E-03	3.17E-03	6.71E-03	2.15E-02
	10D01	**2.71E-03**	5.63E-03	6.51E-03	1.69E-02	3.16E-02
	10D02	1.86E-02	1.64E-02	**1.49E-02**	4.16E-02	4.17E-02
	10D02D	7.17E-03	6.93E-03	**6.13E-03**	1.18E-02	3.22E-02
6	10E01	**1.78E-04**	1.56E-03	7.63E-04	4.27E-03	1.42E-02
	10E02	**1.91E-04**	1.68E-03	1.40E-03	1.06E-02	1.35E-02
	10E03	**2.65E-04**	1.26E-03	1.07E-03	1.75E-03	6.64E-03
	10D01	**2.98E-04**	3.10E-03	3.00E-03	1.09E-02	2.35E-02
	10D02	**4.86E-03**	6.62E-03	5.77E-03	2.61E-02	1.80E-02
	10D02D	**1.02E-03**	3.19E-03	3.14E-03	7.71E-03	1.91E-02
2	10E01	**8.51E-05**	1.10E-03	9.50E-04	1.49E-03	4.83E-03
	10E02	**2.45E-04**	2.36E-03	1.94E-03	5.58E-03	1.13E-02
	10E03	**9.77E-05**	1.16E-03	1.01E-03	1.52E-03	5.25E-03
	10D01	**2.92E-04**	2.46E-03	2.42E-03	2.40E-03	1.48E-02
	10D02D	**2.62E-04**	2.64E-03	2.09E-03	2.44E-03	1.08E-02
1	10E01	**9.05E-05**	1.15E-03	1.62E-03	1.40E-03	2.24E-04
	10E02	**1.53E-04**	1.18E-03	1.66E-03	4.78E-04	2.50E-04
	10E03	**1.13E-04**	1.27E-03	1.80E-03	1.49E-03	2.39E-04
	10D01	**3.58E-04**	2.86E-03	4.10E-03	1.91E-03	5.96E-04
	10D02D	**1.88E-04**	2.56E-03	3.34E-03	2.87E-03	5.24E-04

Table 4: *Optimal parameters selected for surrogate models*

Dimension	Method	Parameter	Output					
			10E01	10E02	10E03	10D01	10D02	10D02D
11	SVR	C	32	8	32	8	8	32
		ϵ	0.0001	0.001	0.0001	0.0001	0.001	0.0001
	ANN	Size of NN	[11 22 1]					
	DBN	Size of RBM	[55 55]					
		BP iterations	3000					
	RF	Number of trees	2000					
		Number of selected features	4					
	RVFL	Number of hidden layer neurons	10000					
6	SVR	C	22.63	32	11.31	32	32	32
		ϵ	0.0001	0.0001	0.0001	0.0001	0.0032	0.0001
	ANN	Size of NN	[6 12 1]					
	DBN	Size of RBM	[30 30]					
		BP iterations	3000					
	RF	Number of trees	2000					
		Number of selected features	2					
	RVFL	Number of hidden layer neurons	4000					
2	SVR	C	22.63	22.63	32	32	N.A.	22.63
		ϵ	0.0001	0.0001	0.0001	0.0001	N.A.	0.0001
	ANN	Size of NN	[2 4 1]					
	DBN	Size of RBM	[10 10]					
		BP iterations	3000					
	RF	Number of trees	2000					
		Number of selected features	1					
	RVFL	Number of hidden layer neurons	1000					
1	SVR	C	8	22.63	16	22.63	N.A.	8
		ϵ	0.0001	0.0001	0.0001	0.0001	N.A.	0.0001
	ANN	Size of NN	[1 2 1]					
	DBN	Size of RBM	[5 5]					
		BP iterations	3000					
	RF	Number of trees	2000					
		Number of selected features	1					
	RVFL	Number of hidden layer neurons	1000					

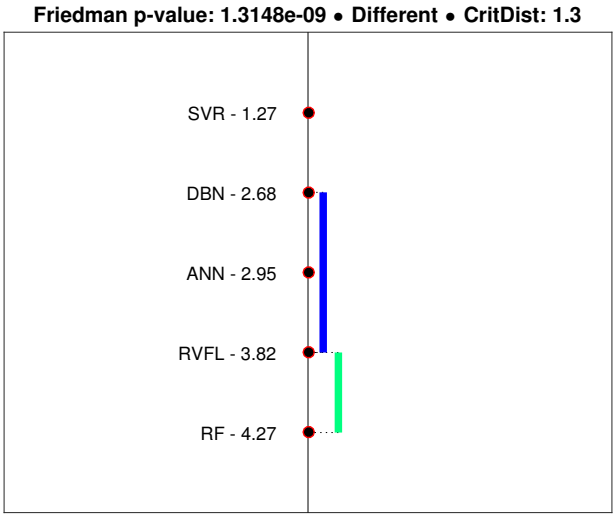


Figure 9: *Nemenyi testing results for surrogate models based on RMSD. The models within a vertical line whose length is less than or equal to a critical distance have statistically the similar performance.*

This can be understood with the following argument. DBN may have resulted in overfitting with the simulation-based data set due to its simpler nature. Moreover, random forest, as a decision tree based ensemble method, has the limitation of accuracy for regression problems due to its dependence on the mean or median of the samples in each leaf node. It is also worth noting that RVFL, as a non-iterative machine learning model with closed-form solutions, constructs surrogate models with reasonable accuracy and high efficiency, which will be showed in Section 4.4.

We use RMSD plots to compare the machine learning surrogates with polynomial and HDMR fitting discussed in [44]. Figure 10 shows the comparative performance among various surrogates for heat duty of reactor 10D01 with respect to all 11 inputs. We consider the polynomial surrogate of 3^{rd} order ($P3$) and HDMR with 2^{nd} order interactions ($H2b$) for this comparison [44]. Figure 10 clearly shows that SVR, ANN and DBN outperform polynomial and HDMR surrogates, while RVFL performs equally.

4.3 Performance comparison using residual plots

We now illustrate the compare performance using residual plots. Figures 11 and 12 show the residual plots for 11-dimensional surrogates of heat duties of reactor 10D01 and heater 10E03 respectively. On the other hand, Figure 13 presents the residual plots for simple 1-dimensional surrogates of 10D01. Since the patterns of the residual plots are similar DBN and ANN, the residual plots only for DBN are shown to simplify the comparison.

From Figures 11 and 12, we can see that all surrogate models do not follow a polynomial relation resulting in non-random distribution of the residuals for 11-dimensional inputs, which proves the effectiveness of machine learning models for constructing the

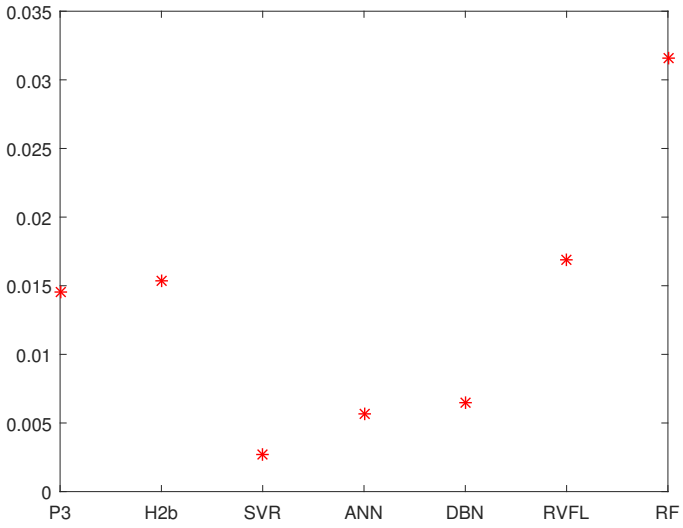
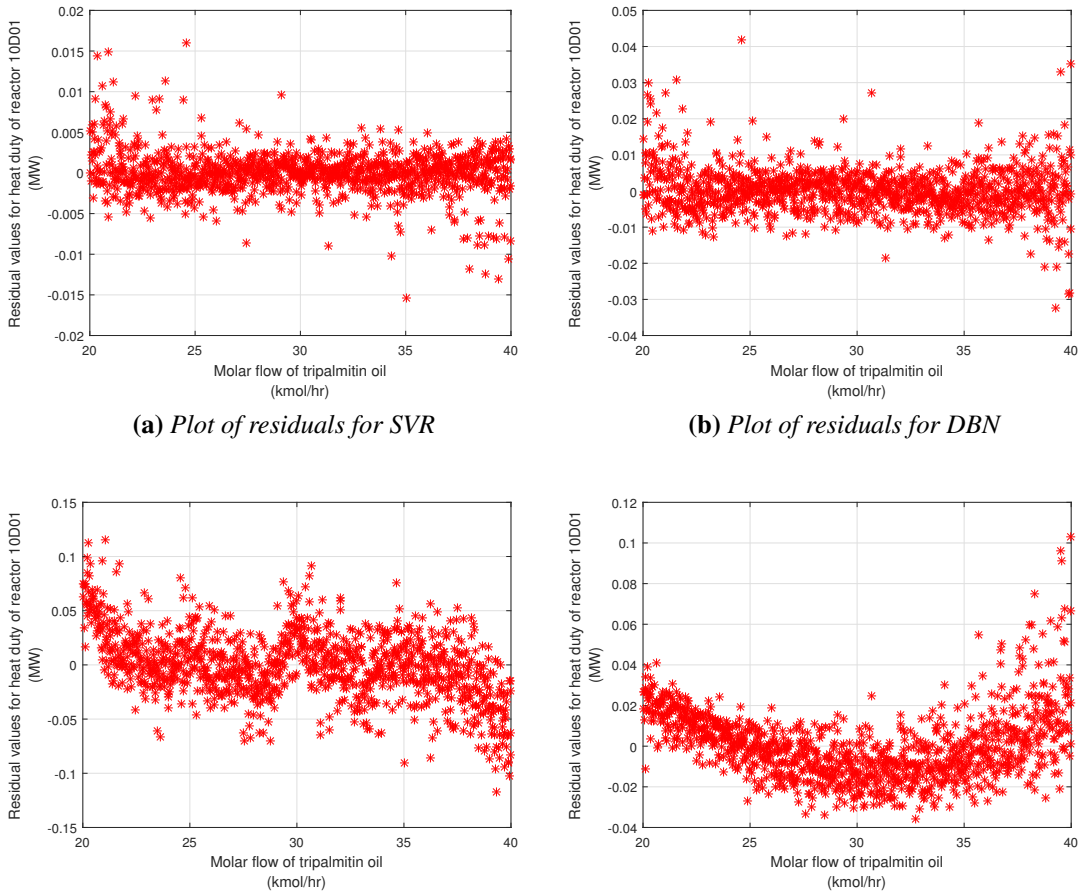


Figure 10: *Plots of RMSD values produce by polynomial fitting, HDMR model and machine learning methods for heat duty of reactor 10D01 with respect to all* 11 *inputs.*

surrogates. However, the plots for RF and RVFL still show some degree of pattern for the distribution of the residues. Meanwhile, the comparison between Figures 11 and 13 shows that the non-random features are much more difficult to identify for surrogates with high dimensional input. Magnitude of residuals in all cases are relatively small indicating strong predictive powers of all the surrogate models. Finally, the residual plots confirm that SVR is one of the best methods for constructing surrogate models, followed by neural networks (e.g. ANN and DBN), while RF shows a stable performance ignoring the input dimension.

4.4 Computation time comparison

Figure 14a shows the computation time of benchmark machine learning methods for constructing 11-dimensional surrogates, while the computation time per evaluation is shown in Figure 14b. Obviously, the computational speed of RVFL is superior than NNs and SVR. SVR requires a grid search on C and ϵ, and NNs are iteratively tuned by BP algorithm to convergence to the optimal weights. These repetitive parameter tuning processes make NNs and SVR less efficient than RVFL that has a closed form solutions. Besides, the RVFL-based surrogate models can easily update the weights according to new input samples [10]. Therefore, RVFL is a good choice when the underlying model is not very complex, and high efficiency is required.



(a) *Plot of residuals for SVR*

(b) *Plot of residuals for DBN*

(c) *Plot of residuals for RF*

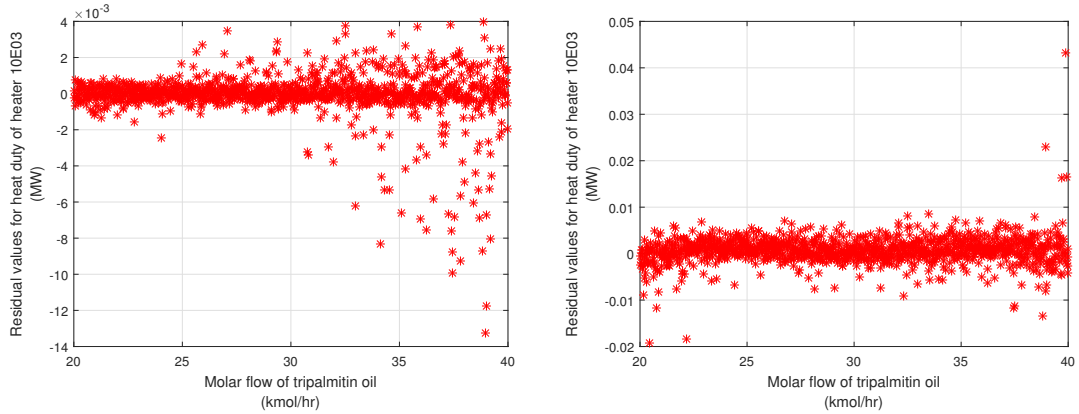
(d) *Plot of residuals for RVFL*

Figure 11: *Plot of residuals against molar flow of tripalmitin oil for heat duty of reactor 10D01 produced for* 11 *inputs.*

5 Conclusions

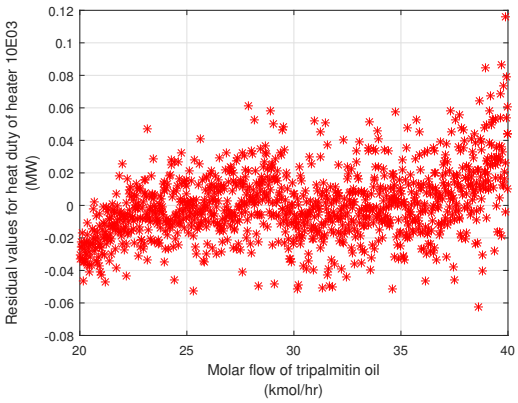
In this paper, various machine learning techniques were investigated and employed for constructing surrogate models to analyze the input-output relations within process flowsheet of a biodiesel plant. The model under investigation includes a reaction and separation steps with the auxiliary equipment and is solved for steady-state operation. A variety of scenarios are considered and 5 different surrogates (support vector regression (SVR), artificial neural network (ANN), deep belief network (DBN), random forests (RF), and random vector functional link network (RVFL)) are used. The performance of surrogate models is evaluated by three error measures: R^2, RMSD and residuals. Based on our analysis and results, following conclusions are made:

1. According to both R^2 (≥ 0.99) and RMSD (with very small values from 10^{-2} to 10^{-4}), all machine learning technique based surrogates achieve a good performance regardless of the domain size and the number of dimensions.

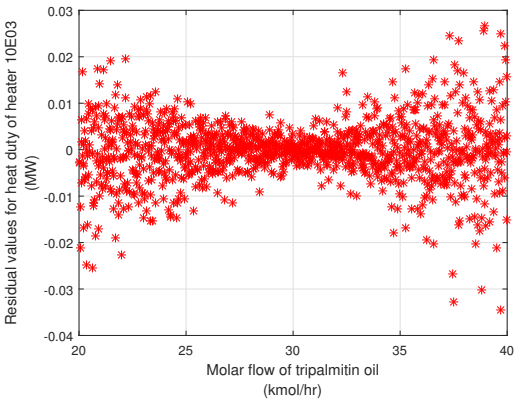


(a) *Plot of residuals for SVR*

(b) *Plot of residuals for DBN*



(c) *Plot of residuals for RF*



(d) *Plot of residuals for RVFL*

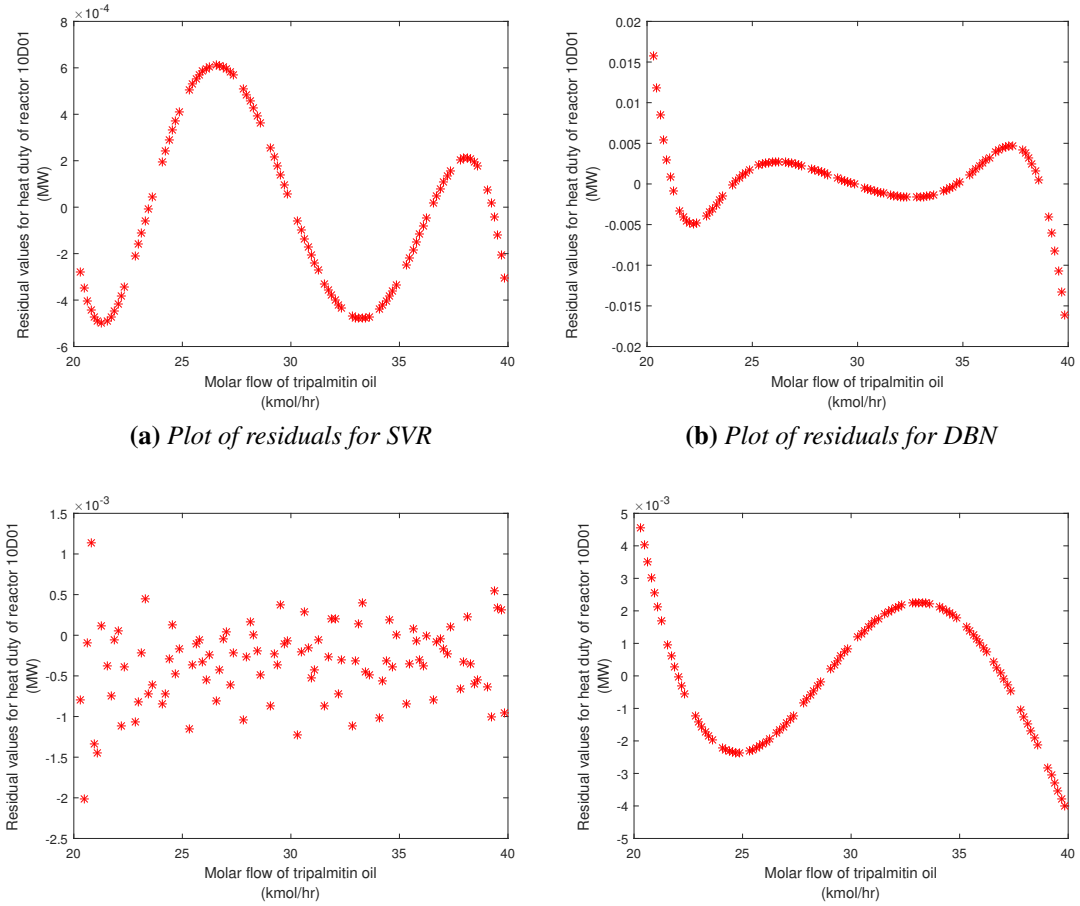
Figure 12: *Plot of residuals against molar flow of tripalmitin oil for heat duty of heater 10E03 produced for* 11 *inputs*

2. SVR achieves the best performance for constructing surrogate models followed by DBN and ANN. These methods outperform polynomial and HDMR, and also captures a random distribution of residuals for multidimensional surrogates.

3. DBN, a deep learning method, has a similar performance as that of ANN, and even performs worse in some cases.

4. Random Forests, a decision tree based method, has the limitation of accuracy for regression problems due to its dependence on the mean or median of the samples in each leaf node. Note that RF achieves relatively random distribution of residuals even for 1-dimensional surrogate.

5. The surrogate models constructed using RVFL have reasonable accuracy and high efficiency due to the reason that RVFL is a non-iterative machine learning model with a closed-form solutions. Hence, RVFL is a good choice for surrogate models when the underlying model is not very complex, and high efficiency is required.



(a) *Plot of residuals for SVR*

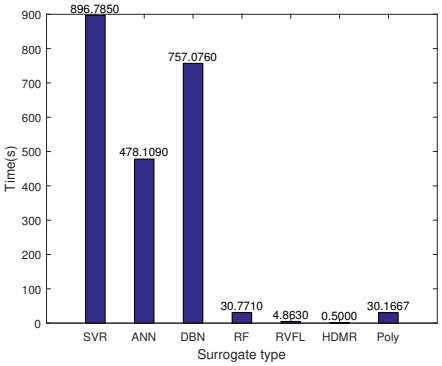
(b) *Plot of residuals for DBN*

(c) *Plot of residuals for RF*

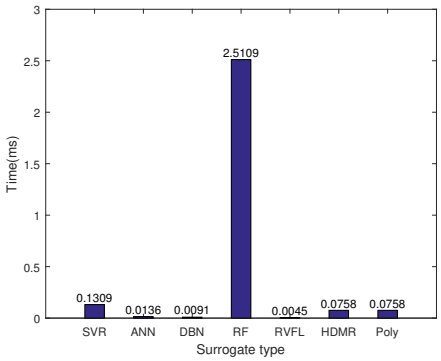
(d) *Plot of residuals for RVFL*

Figure 13: *Plot of residuals against molar flow of tripalmitin oil for heat duty of reactor 10D01 produced for* 1 *inputs.*

In future, more machine learning algorithms will be investigated for constructing surrogates of more complex chemical processes. For example, deep recurrent neural networks will be employed for constructing surrogate models to approximate a hybrid chemical model with feedback loops generated by a number of interconnected models. Moreover, the multi-domain simulations (*e.g.* a combined chemical and electrical engineering system) in the context of the EIPs will also be explored for approximation studies.



(a) *Computation time for training the surrogate model*



(b) *Computation time per evaluation*

Figure 14: *Training and evaluation time of learning models for constructing* 11*-dimensional surrogates of heat duties of reactor 10D01.*

Acknowledgements

This project is funded by the National Research Foundation Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

Nomenclature

Abbreviations

RF: Random Forest
EIP: Eco Industrial Parks
ANN: Artificial Neural Network
SVM: Support Vector Machine
SVR: Support Vector Regression
DBN: Deep Belief Network
RBM: Restricted Boltzmann Machine
SLFN: Single-hidden Layer Feedforward Neural network
RMSD: Root Mean Squared Deviation
RVFL: Random Vector Functional Link
HDMR: High Dimensional Model Representation

References

[1] AspenTech-Aspen Plus v8.6, Mar. 2017. URL http://www.aspentech.com/products/engineering/aspen-plus/.

[2] Microsoft COM technical overview, Mar. 2017. URL https://developer.microsoft.com/en-us/windows/desktop.

[3] P. Azadi, G. P. Brownbridge, S. Mosbach, O. R. Inderwildi, and M. Kraft. Production of biorenewable hydrogen and syngas via algae gasification: A sensitivity analysis. *Energy Procedia*, 61:2767–2770, 2014.

[4] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2:1–127, Jan. 2009.

[5] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[6] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[7] L. Breiman, J. H. Freidman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis, 1984. ISBN 9780412048418.

[8] G. Brundtland, M. Khalid, S. Agnelli, S. Al-Athel, B. Chidzero, and L. Fadika. *Our Common Future: The World Commission on Environment and Development*. Oxford University Press, 1987.

[9] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[10] C. Chen and J. Wan. A rapid learning and dynamic stepwise updating algorithm for flat neural networks and the application to time-series prediction. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(1):62–72, 1999.

[11] N. Chen, K. Wang, C. Xiao, and J. Gong. A heterogeneous sensor web node meta-model for the management of a flood monitoring system. *Environmental Modelling & Software*, 54:222–237, 2014.

[12] M. R. Chertow. Industrial symbiosis: Literature and taxonomy. *Annual Review of Energy and the Environment*, 25:313–337, 2000.

[13] E. Cimren, J. Fiksel, M. E. Posner, and K. Sikdar. Material flow optimization in by-product synergy networks. *Journal of Industrial Ecology*, 15(2):315–332, 2011.

[14] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[15] S. B. Crary. Design of computer experiments for metamodel generation. *Analog Integrated Circuits and Signal Processing*, 32(1):7–16, 2002.

[16] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[17] P. Desrochers. Cities and industrial symbiosis: Some historical perspectives and policy implications. *Journal of Industrial Ecology*, 5(4):29–44, 2001. ISSN 1530-9290.

[18] N. Draper and H. Smith. *Applied regression analysis, 3rd Edition*. Wiley series in probability and statistics: Texts and references section. Wiley, 1998.

[19] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. *Advances in neural information processing systems*, 9:155–161, 1997.

[20] A. Forrester, A. Sóbester, A. Keane, A. I. of Aeronautics, and Astronautics. *Engineering Design Via Surrogate Modelling: A Practical Guide*. Progress in astronautics and aeronautics. J. Wiley, 2008.

[21] P. Geyer and A. Schlüter. Automated metamodel generation for design space exploration and decision-making a novel method supporting performance-oriented building design and retrofitting. *Applied Energy*, 119:537–556, 2014.

[22] S. Haykin. *Neural Networks: A Comprehensive Foundation*. International edition. Prentice Hall, 1999. ISBN 9780132733502.

[23] G. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[24] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, Aug. 2002.

[25] G. E. Hinton. *Neural Networks: Tricks of the Trade: Second Edition*, chapter A Practical Guide to Training Restricted Boltzmann Machines, pages 599–619. Springer Berlin Heidelberg, 2012.

[26] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, Jul. 2006.

[27] T. K. Ho. Random decision forests. In *Proc. the Third International Conference on Document Analysis and Recognition*, pages 278–282, Montreal, Que, Aug. 1995.

[28] T. K. Ho. The random subspace method for constructing decision forests. *Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[29] C. Hoffman. *The Industrial Ecology of Small and Intermediate-sized Technical Companies: Implications for Regional Economic Development*. Program on the Role of Growth Centers in Regional Economic Development: Discussion Paper. Center for Economic Development, University of Texas, 1971.

[30] R. J. Hyndman and A. B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22:679–688, 2006.

[31] I. Kantor, M. Fowler, and A. Elkamel. Optimized production of hydrogen in an eco-park network accounting for life-cycle emissions and profit. *International Journal of Hydrogen Energy*, 37:5347–5359, 2012.

[32] M. Karlsson. The mind method: A decision support for optimization of industrial energy systems principles and case studies. *Applied Energy*, 88:577–589, 2011.

[33] C. A. Kastner, A. Braumann, P. L. Man, S. Mosbach, G. P. Brownbridge, J. Akroydand, M. Kraft, and C. Himawan. Bayesian parameter estimation for a jet-milling model using metropolishastings and wanglandau sampling. *Chemical Engineering Science*, 89:244–257, 2013.

[34] J. P. Kleijnen. Kriging metamodeling in simulation: A review. *European Journal of Operational Research*, 192:707–716, 2009.

[35] C.-N. Ko and C.-M. Lee. Short-term load forecasting using SVR (support vector regression)-based radial basis function neural network with dual extended kalman filter. *Energy*, 49:413–422, 2013.

[36] Z. W. Liao, J. T. Wu, B. B. Jiang, J. D. Wang, and Y. R. Yang. Design methodology for flexible multiple plant water networks. *Industrial & Engineering Chemistry Research*, 46(14):4954–4963, 2007.

[37] R. B. Palm. Prediction as a candidate for learning deep hierarchical models of data. Master's thesis, Technical University of Denmark, 2012.

[38] M. Pan, J. Sikorski, C. A. Kastner, J. Akroyd, S. Mosbach, R. Lau, and M. Kraft. Applying industry 4.0 to the jurong island eco-industrial park. *Energy Procedia*, 75: 1536–1541, 2015.

[39] Y.-H. Pao, S. M. Phillips, and D. J. Sobajic. Neural-net computing and the intelligent control of systems. *International Journal of Control*, 56:263–289, 1992.

[40] Y.-H. Pao, G.-H. Park, and D. J. Sobajic. Learning and generalization characteristics of the random vector functional-link net. *Neurocomputing*, 6:163–180, 1994.

[41] Y. Ren, X. Qiu, P. N. Suganthan, N. Srikanth, and G. Amaratunga. Detecting wind power ramp with random vector functional link (rvfl) network. In *Proc. IEEE Symposium Series on Computational Intelligence (CIEL2015)*, Cape Town, South Africa, Dec. 2015.

[42] Y. Ren, P. N. Suganthan, N. Srikanth, and G. Amaratunga. Random vector functional link network for short-term electricity load demand forecasting. *Information Sciences*, 000:1–16, 2016.

[43] E. Roux and P.-O. Bouchard. Kriging metamodel global optimization of clinching joining processes accounting for ductile damage. *Journal of Materials Processing Technology*, 213:1038–1047, 2013.

[44] J. J. Sikorski, G. Brownbridge, S. S. Garud, S. Mosbach, I. A. Karimi, and M. Kraft. Parameterisation of a biodiesel plant process flow sheet model. *Computers & Chemical Engineering*, 95:108–122, 2016.

[45] T. Simpson, J. Poplinski, P. N. Koch, and J. Allen. Metamodels for computer-based engineering design: Survey and recommendations. *Engineering with Computers*, 17 (2):129–150, 2001.

[46] H. A. Song and S.-Y. Lee. Hierarchical representation using NMF. *Neural Information Processing*, 8226:466–473, 2013.

[47] T. Yamashita, M. Tanaka, E. Yoshida, Y. Yamauchi, and H. Fujiyoshi. To be Bernoulli or to be Gaussian, for a restricted Boltzmann machine. In *22nd International Conference on Pattern Recognition*, pages 1520–1525, 2014.

[48] L. Zhang and P. N. Suganthan. Random forests with ensemble of feature spaces. *Pattern Recognition*, 47:3429–3437, 2014.

[49] L. Zhang and P. N. Suganthan. Oblique decision tree ensemble via multisurface proximal support vector machine. *IEEE Transactions on Cybernetics*, 45(10):2165–2176, 2015.

[50] L. Zhang and P. N. Suganthan. A comprehensive evaluation of random vector functional link networks. *Information Sciences*, 367-368:1094–1105, 2016.

[51] L. Zhang and P. N. Suganthan. A survey of randomized algorithms for training neural networks. *Information Sciences*, 000:1–10, 2016.