Soft-sensor development for biochemical systems using genetic programming

Suraj Sharma, Sanjeev S. Tambe*

Artificial Intelligence Systems Group, Chemical Engineering and Process Development Division, National Chemical Laboratory, Dr. Homi Bhabha Road, Pune 411008, India

A R T I C L E   I N F O

Article history:
Received 20 August 2013
Received in revised form 5 February 2014
Accepted 6 February 2014
Available online 15 February 2014

Keywords:
Artificial intelligence
Batch processing
Bioprocess monitoring
Lipase
Modeling
Sensors

A B S T R A C T

Soft-sensors are software based process monitoring systems/models. In real-time they estimate those process variables, which are difficult to measure online or whose measurement by analytical procedures is tedious and time-consuming. In this study, the genetic programming (GP), an artificial intelligence based data-driven modeling formalism, has been introduced for the development of soft-sensors for biochemical processes. The novelty of the GP is that given example input–output data, it searches and optimizes both the form (structure) and parameters of an appropriate linear/nonlinear data-fitting model. In this study, GP-based soft-sensors have been developed for two bioprocesses, namely extracellular production of lipase enzyme and bacterial production of poly(3-hydroxybutyrate-co-3-hydroxyvalerate) copolymer. While in case study-I, the soft-sensor predicts the time-dependent lipase activity (U/ml), in case study-II it predicts the amount of accumulated polyhydroxyalkanoates (% dw). The prediction and generalization performance of the GP-based soft-sensors was compared with the corresponding multi-layer perceptron (MLP) neural network and support vector regression (SVR) based soft-sensors. This comparison indicates that in the first case study the GP-based soft-sensor with the training and test set correlation coefficient (root-mean-squared-error) magnitudes of ∼0.96 (∼0.962 U/ml) has clearly outperformed the two other soft-sensors. In case study-II involving bacterial copolymer production, the GP and SVR based soft-sensors have performed equally well (correlation coefficient ∼0.98) while the MLP based soft-sensor’s performance was relatively inferior (correlation coefficient ∼0.94).

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

In today’s industrial world, various types of sensors are needed to provide speedy and reliable measurements of a wide variety of process and/or product related variables and parameters. Sensors are sophisticated devices used in detecting and producing a measurable response to a change in, for example, chemical, physical, electrical, biochemical or optical state of a system. These measurements assist process operators and engineers in: (i) knowing the current state of the process, (ii) controlling and monitoring of the process, (iii) detecting and diagnosing any abnormal process behavior timely, (iv) taking corrective actions in the event of an abnormal process behavior, and (v) optimizing the performance of the process with a view to minimize costs and/or improve its efficiency. In many instances, however, an appropriate hardware-based robust sensor for measuring a process variable is either unavailable or the alternative analytical procedure is time-consuming and tedious. In such situations, the alternative of developing a soft-sensor should be explored. A soft-sensor is a software module capable of estimating a process variable in real-time. This module comprises a mathematical model, which makes use of the available quantitative knowledge regarding other process variables and parameters to estimate the magnitude of the chosen variable. The available information pertaining to other variables could be in the form of sensor measurements and/or mathematical models.

The challenges involved in the soft-sensor development for biochemical processes are the same as encountered in their modeling, optimization and control; the notable ones are as given below.

- Bioprocesses are characterized by their complex dynamics, such as inverse response, dead time and strong nonlinearities. These stem primarily from their main driving force, namely, microorganisms (cells), which are very sensitive to any variations in
the reaction environment (e.g., temperature, substrate concentration, pH, among others) [1].

• An important class of bioprocesses, i.e., batch fermentation, commonly evolves through three stages, namely lag, exponential and stationary stages. The factors that influence the behavior of micro-organisms vary in each stage, owing to which the batch fermentation system exhibits different nonlinear characteristics in different stages. As a result, a global soft-sensor model for batch fermentation leads to complicated structure with limited prediction accuracy [2].

• Crucial biochemical variables and/or parameters are hard to measure online in bioprocesses such as batch fermentation.

• In bioprocesses involving induced cultures, there exists a variation in the morphology, energy metabolism and macroscopic composition of the cells; hence quantification of “biomass” or similar variables is not straightforward [3].

Since 1970–1980s the cost of computer-based instrumentation lowered significantly, and the concept of soft-sensor gained ground in the process estimation and inferential controls [4], bioprocess monitoring [5,6], control of nonlinear bioprocesses [7], biological wastewater treatment [8], melt index prediction [9], etc. For developing a soft-sensor, two principal approaches are phenomenological and empirical modeling. The former approach is employed when the detailed knowledge about the physico-chemical phenomena (kinetics, mass transfer, thermodynamics, etc.) underlying the process is available. Very often, gaining this knowledge itself becomes a tedious and costly task owing to the complex nature of the process and the extensive experimentation involved in collecting the necessary data. These difficulties make the phenomenological modeling route to soft-sensor development impractical. In such a situation, empirical modeling can be resorted to for the development of a soft-sensor.

There exist three commonly utilized methodologies for developing empirical models, namely regression analysis, artificial neural networks (ANNs) and support vector regression (SVR). For a pre-specified data-fitting function, the linear/nonlinear regression estimates the magnitudes of the function parameters that fit the given input–output data. Since many chemical and biochemical processes exhibit nonlinear behavior choosing an appropriate data-fitting function from a large number of possible alternatives becomes a daunting task. Despite expending a huge effort in guessing and testing different nonlinear data-fitting functions, there is no guarantee that a well-fitting function can indeed be secured in a finite number of trials. The other two empirical modeling formalisms, viz. ANNs and SVR, overcome the difficulties associated with the regression analysis since they do not require specification of the exact form of the data fitting function. Accordingly, ANN and SVR formalisms have been exploited in the development of soft-sensors and related applications including control of a distillation process [10], fed-batch reactor operation [11], and hybrid modeling of fermentation process [12]. Although these are potent nonlinear function approximation methods with a wide applicability, the ANNs and to some extent the SVR generate “black box” models whose structure and parameters do not provide any insight into the phenomena underlying the process being modeled.

In the present study, an artificial intelligence (AI) based exclusively data-driven modeling paradigm known as genetic programming (GP) [13] has been proposed for developing soft-sensor models for biochemical processes. Given multiple input–single output (MISO) data, the novelty of the GP formalism lies in its ability to search and optimize the form as also parameters of an appropriate linear/nonlinear data-fitting function. Despite its novelty and attractive properties, the GP formalism has not been explored widely for data-driven modeling applications in chemical and biochemical sciences/engineering to the same extent as other exclusively data-driven modeling methods, namely ANNs and SVR. In one of the soft-sensor applications involving the GP formalism, Kordon et al. [14] developed a soft-sensor for the emission estimation in one of the Dow Chemical Company plants in Freeport, TX. In this study, the soft-sensor was developed by integrating three computational intelligence approaches, namely, GP, analytical neural networks, and support vector machines. A rigorous literature survey indicates that the present study is the first one, wherein the GP formalism has been utilized for the development of soft-sensors for biochemical processes. The efficacy of the GP-based soft-sensors for biochemical processes has been demonstrated by conducting two case studies involving microorganism assisted extracellular production of lipase and production of bacterial poly(3-hydroxybutyrate-co-3-hydroxyvalerate) copolymer. In these case studies, multiple input–single output (MISO) example data sets have been utilized in searching and optimizing the functional form (structure) as also parameters of the MISO data-fitting functions (soft-sensors). While in the first case study, the soft-sensor predicts the time-dependent lipase activity (U/ml), in the second case study it predicts the amount of accumulated polyhydroxalkanoates (% dw).

The prediction accuracy and generalization capability of the GP-based soft-sensors have been compared with those developed using the ANN and SVR formalisms.

This paper is structured as follows. Section 2 provides a detailed description of the GP formalism and its implementation. The commonly used feed-forward artificial neural network, namely multilayer perceptron (MLP) and the machine learning based SVR formalism have been described in sections three and four, respectively. The two case studies wherein the GP-based soft-sensor models have been developed for two biochemical systems are presented in Section 5. This section also provides results of the comparison of the GP, MLP and SVR based soft-sensor models pertaining to the two biochemical systems. Finally, Section 6 summarizes the principal findings of the study.

2. Genetic programming (GP)

In its original form, the GP formalism was proposed as a method for automatically generating computer programs that perform predefined tasks [13]. It is an extension of the Genetic algorithm (GA) formalism [15]. Given an objective function, the GA efficiently searches and optimizes the values of the decision variables that would maximize or minimize the function. Similar to the GA, the GP is founded on the Darwinian principles of natural selection and reproduction. Accordingly, the GP implementation uses simplified analogs of the naturally occurring genetic operations namely, crossover and mutation. There exist a number of schemes for implementing the genetic programming methodology, such as the tree-structured GP, linear GP, gene expression programming, multi expression programming, grammatical evolution, Cartesian GP and stack-based GP. Among these the tree-structured GP forms the most commonly employed GP-implementation.

Apart from automatically generating computer codes that execute pre-specified tasks, the GP can also be used for an attractive and novel application known as symbolic regression. Given an example data set comprising independent (predictor/causal) and dependent (response) variables, the GP-based symbolic regression is capable of searching and optimizing the form (structure) and associated parameters of a suitable linear/nonlinear mathematical model that fits the data—or at least an approximation to these. A notable feature of the GP-based symbolic regression is that unlike ANNs and SVR, it makes no assumptions about the form of the data-fitting function. The unique benefits of the symbolic regression include a human insight into and interpretability of the obtained models, identification of the key variables and
their combinations in the data, and generation of computationally simple expressions for easy deployment into operational models [16]. In a noteworthy study, Schmidt and Lipson [17] have demonstrated that the symbolic regression can be employed to search the “natural law” underlying a physical phenomenon (pendulum dynamics). Other applications of the GP include bioprocess monitoring [18], fermentation modeling [19], electronic nose [20], synthesis of heat-integrated complex distillation systems [21], classification of Raman spectra [22], and optimization of a controlled release pharmaceutical formulation [23].

2.1. GP based symbolic regression

Given a multiple input–single output (MISO) example dataset, the task of the GP-based symbolic regression is to fit the following model to the data (also see Fig. 1):

\[ y = f(x, \alpha) \]  

where, \( y \) represents the dependent variable, \( x = [x_1, x_2, \ldots, x_N]^T \) constitutes an \( N \)-dimensional vector of independent variables (model inputs), \( f \) represents a linear/nonlinear data-fitting function whose form, and parameters, \( \alpha = [\alpha_1, \alpha_2, \ldots, \alpha_M]^T \), are to be searched and optimized by the GP. The generic step-wise GP-based symbolic regression procedure for addressing the data-fitting problem described above is given below (also see the GP flowchart in Fig. 2).

2.1.1. GP-implementation

Consider an MISO example data set, \( D = \{x_i, y_i\}, i = 1, 2, \ldots, N_p \), consisting of \( N_p \) number of input–output patterns available for conducting the GP-based model development. Each input pattern \( x \) is \( N \)-dimensional and the corresponding output \( y \) is a scalar.

Step (a) (Initialization of population of candidate solutions): To begin, the GP procedure randomly generates a pre-specified number of candidate solutions (mathematical expressions) to the given data-fitting problem. Specifically, a set, \( F \), of \( N_{pop} \) number of functions defining the initial random population of candidate solutions is generated:

\[ F = \{f_j\}; j = 1, 2, \ldots, N_{pop} \]  

Each candidate solution \( f_j \) in the population is coded in the form of a hierarchical tree structure as illustrated in Fig. 3. In the tree structure, two types of nodes, namely operator and operand, are used for coding an expression. The operator node (also termed “function” node) represents a mathematical operator; a few of the possible operators are as given below:

- Arithmetic operators: addition, subtraction, multiplication and division
- Trigonometric and other: \( \sin, \cos, \log, \exp \), etc.

An operand (also termed “terminal”) node represents a model input (element of \( x \)), or a function parameter (element of \( \alpha \)). An illustration of a symbolically coded tree-structure defining \( f = (\alpha_1 \times x_1) + (\alpha_3 - x_2)/\alpha_2^2 \) is portrayed in Fig. 3.

Step (b) (Fitness evaluation): Fitness value (fitness score) of a candidate solution/tree structure is a measure of the data-fitting potential of the solution and it can be evaluated using the following mathematical expression:

\[ R_j = \frac{1}{1 + \Delta_j}; \quad j = 1, 2, \ldots, N_{pop} \]  

Here \( R_j \) is the fitness value of the \( j \)th candidate solution and \( \Delta_j \) is the mean squared error (MSE) evaluated using,

\[ \Delta_j = \frac{\sum_{i=1}^{N_p} (y_i - \hat{y}_{ij})^2}{N_p} \]  

where \( y_i \) is the desired (target) output value pertaining to the \( i \)th input data pattern in the example set and \( \hat{y}_{ij} \) is the model predicted output value when \( i \)th data pattern is used to evaluate the output of the \( j \)th candidate solution. After evaluating their fitness values, candidate solutions in the population are ranked in the decreasing order of their fitness values.

Step (c) (Selection): Inspired by the Darwinian concepts of natural selection (also known as survival of the fittest) and genetic propagation of characteristics, the candidate solutions from the
current population are selected in a specific way to enter the mating pool. In particular, the selection is conducted in a manner such that while the fitter solutions have a higher probability of entering the mating pool, there is also a small probability of selecting lower-fitness solutions. There exist several methods of selection such as the Roulette-wheel selection, tournament selection, and elitist mating [24].

Step (d) (Crossover): In this step, the tree-structured members of the mating pool are used to generate new candidate solutions (offspring). This operation can be implemented in a number of ways. For instance, the single-point crossover illustrated in Fig. 4, involves: (i) choosing randomly two candidate solutions (parents) from the mating pool, (ii) splicing each parent tree at a randomly selected node into two segments, and (iii) mutually exchanging and recombining the spliced segments to produce two offspring trees. This step is repeated till $N_{ oppos}$ numbers of offspring are generated.

Step (e) (Mutation): Mutation comes in two flavors: “node” and “branch” mutations. The former is illustrated in Fig. 5(a) wherein a randomly chosen operator (operand) node is replaced by randomly selected another operator (operand) node. In the branch mutation (see Fig. 5(b)), a randomly chosen sub-tree of an offspring is replaced with a randomly generated tree or a sub-tree. Mutation helps in maintaining the genetic diversity in a population, i.e. it prevents solutions from becoming too similar to each other. Typically, mutation is executed with a small probability. The offspring population emerging from the mutation operation defines a new generation of candidate solutions.

Step (f) (Termination): Repeat steps (b) to (e) until either of the following two termination criteria is satisfied.

- Pre-specified number of generations has been evolved.
- A negligible or no change in the fitness value of the best candidate solution in the offspring population over a number of successive generations.

3. Artificial neural networks

The basic MLP structure portrayed in Fig. 6 is composed of three layers, namely input, hidden and output layers consisting of $N$, $M$ and L processing elements (also termed “nodes” or “neurons”), respectively (where $L = 1$). Given the data set $D$, containing $N_p$ measurements of the input (independent/causal/predictor)–output (dependent/response) variables, the MLP learns the nonlinear interrelationships existing between them by appropriately adjusting the inter-node connection weights. The objective of the weight adjustment is to minimize an error function that represents an appropriate measure of the error between the desired (target) and MLP-predicted outputs. The MLP neural network uses a generic nonlinear transfer function (such as logistic/tanh sigmoid) to compute the outputs of its hidden layer nodes, which imparts it with the nonlinear function approximation capability. The most commonly used method to train the weights of an MLP is the generalized delta rule based error-back-propagation (EBP) algorithm [25]. In the EBP-based training, the network weights are adjusted iteratively such that an error function such as the mean squared error (MSE) is minimized. The details of the MLP training procedure and the various issues involved in obtaining an optimal MLP model can be found, for example, in Tambe et al. [26] and Bishop [27]. The ANNs are referred to as “black-box” models, since their performance is dependent on the quality and size of the data set used in their training as also on the structure of the model. Despite their drawbacks, a number of studies have utilized ANNs in the development of soft-sensors [28–34] owing to their efficacy in conducting exclusively data-based nonlinear modeling.

4. Support vector regression (SVR)

In recent years, support vector regression (SVR) [35,36] has gained a widespread acceptance in the construction of data-driven nonlinear models. It is an adaptation of the statistical/machine learning theory based classification paradigm namely, support vector machines [35]. The SVR possesses some desirable characteristics such as, good generalization ability of the regression function, robustness of the solution, sparseness of the regression and an automatic control of the solution complexity. Moreover, the formalism provides an explicit knowledge of the data points, which are important in defining the regression function. This feature allows an interpretation of the SVR-approximated model in terms of the training data.

Given the example data set $D = \{(x_i, y_i), i = 1, 2, \ldots, N_p\}$, where $x_i$ is a vector of input variables and $y_i$, the corresponding scalar output (target), the objective of an SVR model is to fit a regression function, $y = f(x)$, such that it accurately predicts the outputs ($y_i$) corresponding to a new set of input examples ($x_i$). In SVR, the inputs are first nonlinearly mapped into a high dimensional feature space ($\Phi$) wherein they are correlated linearly with the outputs. The SVR algorithm attempts to place a tube around the regression function as shown in Fig. 7. The region enclosed by the tube is called a $\varepsilon$-insensitive zone, where $\varepsilon$ represents the radius of the tube. The diameter of the tube should ideally be the amount of noise in the data. The optimization criterion in SVR penalizes those data points, whose $y$ values lie more than $\varepsilon$ distance away from the regression function $f(x)$. A detailed description of the SVR and its implementation is can be found in e.g., Vapnik [35], Zaid [37] and Nandi et al. [38]. The SVR-based regression function has the following form

$$f(x, w) = f(x, \alpha, \alpha^*) = \sum_{i=1}^{P} (\alpha_i^* - \alpha_i) K(x_i, x) + b$$

(5)

where, $\alpha_i$ and $\alpha_i^*$ ($\geq 0$) are the coefficients (Lagrange multipliers) satisfying $\alpha_i\alpha_i^* = 0$; $i = 1, 2, \ldots, P$, and $K(x_i, x)$ denotes the kernel function describing the dot product in the feature space. The vector $w$ is described in terms of the Lagrange multipliers $\alpha$ and $\alpha^*$. In Eq. (5), only some of the coefficients, $(\alpha_i^* - \alpha_i)$, are non-zero and the
corresponding input vectors, \( \mathbf{x}_i \), are called “support vectors (SVs).” The SVs can be thought of as the most informative data points that compress the information content of the training set.

5. Case studies

This section provides details of the development of GP-based soft-sensor models for two biochemical processes, namely microorganism assisted extracellular production of lipase and production of a bacterial copolymer. While in case study-I, the soft-sensor predicts the time-dependent lipase activity (U/ml), in case study-II the soft-sensor predicts the amount of accumulated polyhydroxyalkanoates (% dwc). The prediction performance of the GP-based soft-sensors has been compared with those developed using the MLP neural network and SVR methodologies.

In data-driven modeling, it is important to avoid “over-fitting” of a model. It occurs when a model is over-trained (i.e., trained over a large number of iterations or generations) or over-parameterized (the fitting function contains more terms and parameters than necessary). Over-fitting tends to increase the complexity of the fitted model and performs poorly at generalization. To avoid over-fitting, the “single train/test split” method [17] has been employed in this study. In this method the available input–output example set is divided into two sets namely training and test sets. While the former is used to train the model, the test set is used for evaluating the generalization performance of the model. The model predicting the test set outputs with a high and comparable (in relation to the training set output predictions) accuracy is considered to possess good generalization ability and therefore accepted. Once a GP-based soft-sensor model with good prediction accuracy and generalization capability is secured, its parameters, \( \alpha \), can be further fine-tuned by using the Levenberg–Marquardt nonlinear regression technique [39].

For developing the GP-based soft-sensor, a software package known as Eureqa Formulize (version 0.98.01) [17] was utilized. This package has been optimized to find parsimonious models (i.e., trees of small size), which are expected to possess good generalization ability. The Eureqa Formulize forms the initial expressions by randomly combining mathematical building blocks such as algebraic operators \( \{+,-,/,\times\} \), analytical functions (sine, cosine, tan, etc.), constants, and state variables. This software employs the simple “single train/test split” procedure (and not the more rigorous “k-fold” cross-validation) wherein a fixed training set and a test set is utilized for assessing the prediction accuracy and generalization performance of a candidate expression. There exists a number of options in the Eureqa package for data preprocessing and generation of expressions; the options available for the latter task are listed in Appendix A.

In each case study, several runs of the GP-based symbolic regression were performed by using each time a different set of operators from among the large set of operators listed in Appendix A. The converged soft-sensor model obtained in each trial was evaluated in terms of the following statistical quantities (computed for both training and test set data) [40]: (i) the root-mean-squared-error (RMSE) between the desired (experimental/target) and soft-sensor predicted values of the model output, (ii) coefficient of correlation \( R \) between the desired and soft-sensor predicted output values, and (iii) coefficient of determination (variance) \( R^2 \) between the desired and soft-sensor predicted output values. Only those soft-sensor models, which possessed following attributes were accepted as the feasible ones: (i) models consisting of all the specified input variables, (ii) models with minimum complexity (small number of terms and parameters), (iii) low and comparable magnitudes of RMSE for both training and test set data, and (iv) high and comparable magnitudes of \( R \) in respect of the desired and soft-sensor model predicted output values in the training and test sets.

5.1. Case study-I: soft-sensor for extracellular production of lipases

Lipase—a proteolytic enzyme—belongs to the group of hydro-lases that catalyzes the breakdown of fats and oils with the subsequent release of fatty acids and glycerols. In addition, lipases catalyze synthesis, hydrolysis, and trans-esterification of esters and
(a) Node Mutation:

(b) Branch Mutation:

Exhibit enantioselective properties [41]. Lipase performs an important role in the digestion, transport, and processing of dietary lipids that are found in animals, plants, and microorganisms [42]. The ability to perform a specific chemical transformation (biotransformation) has made the lipase an increasingly popular enzyme in industries such as food, detergent, cosmetics, organic synthesis, and pharmaceuticals [43,44]. The microbial routes of lipase production have gained a special industrial attention due to their stability, selectivity, and broad substrate specificity [45,46]. A number of potential microorganisms for the extracellular production of lipases have been reviewed in [47].

Burkert et al. [48] have studied and optimized the extracellular production of lipase using Geotrichum candidum fungus. They examined the effect of nutrients composed of oil (Olive oil (OO)/Soy oil (SO)) as a carbon source and ammonium nitrate (NH$_4$NO$_3$) and corn steep liquor (CSL) as nitrogen sources for the production of extracellular lipases (U/ml) ($y_L$). The progress of the reaction was monitored at different intervals of fermentation time. The measurement of $y_L$ involves various analytical laboratory procedures such as titrimetric, fluorometric, calorimetric, turbidimetric, chromatographic, radiometric, enzymatic, physical and immunological analyses. Majority of these procedures due to their high cost and time requirements are not yet suitable for the non-purified samples and for conducting large scale analyses [49]. Various types of biosensors, such as electrochemical, optical, mechanical and thermal, are available for determining the lipase activity and quantity of its substrates and these have been reviewed in [50]. The response time of these biosensors varies between 0.3 and 20 min. Since speedy estimation of $y_L$ greatly assists in monitoring and controlling the reaction efficiently and maximizing the product yield, it is desirable to develop a soft-sensor that almost instantaneously predicts the lipase production. Accordingly, a GP-based soft-sensor has been developed, which as a function of three process variables and fermentation time, predicts the magnitude of the extracellular lipase activity, $y_L$ (U/ml) at the instant defined by the fermentation time.

Fig. 5. Illustration of node and branch mutations.
5.1.1. Results and discussion

The experimental data set used in the development of the GP-based soft-sensor for predicting the lipase activity, \( y_L \), was sourced from Burkert et al. [48]. The four process variables namely, soy oil (SO) concentration (% v/v) (\( x_1 \)), NH\(_4\)NO\(_3\) concentration (\( x_2 \)) (% w/v), CSL concentration (\( x_3 \)) (% v/v), and fermentation time (\( x_4 \)) (hours), form inputs to the soft-sensor model. The example set for the soft-sensor development consisted of data pertaining to 17 experiments wherein each input–output variable was monitored at ten time intervals of fixed length. This data set was rearranged into 170 input–output patterns portraying hourly readings. The said experimental data are listed in Supplementary Table 1. The Eureqa package partitioned the example set in the user-specified 70:30 ratio as a result of which the training and test sets respectively contained 119 and 51 input–output data patterns; In Supplementary Table 1, the test patterns are marked by “***” symbol.

The GP-based optimal soft-sensor model generated by the Eureqa package upon following the symbolic regression procedure described earlier is given as:

\[
\hat{y}_L = 0.6428\hat{x}_3 + \frac{(\hat{x}_1 - \hat{x}_4)}{(-6.164 - \hat{x}_1 - 6.767\hat{x}_3)} + \frac{(0.2818\hat{x}_4 - 0.1943 - \hat{x}_1)}{(-4.491 - 2.759\hat{x}_3)}
\]

(6)

where \( \hat{y}_L, \hat{x}_1, \hat{x}_2, \hat{x}_3 \) and \( \hat{x}_4 \), respectively represent the normalized values of variables, \( y_L, x_1, x_2, x_3 \) and \( x_4 \). The normalized variables were obtained as follows:

\[
\hat{x}_i = \frac{x_i - \bar{x}_i}{s_j}; \quad i = 1, 2, \ldots, N_p; \quad j = 1, 2, \ldots, N
\]

(7)

where \( x_i \) represents ith value of jth un-normalized variable \( x_j \); \( \bar{x}_i \) refers to mean of \( x_i \), and \( s_j \) represents standard deviation of \( x_i \). Similar procedure was employed for the normalization of \( y_L \). The mean and standard deviation values used in the normalization are:

\[
\tilde{x}_1 = 1.08\% (v/v); \quad \tilde{x}_2 = 1(\%, w/v); \quad \tilde{x}_3 = 7.25(\%, v/v); \quad \tilde{x}_4 = 33.131(\text{h}); \quad \tilde{y}_L = 13.162(\text{U/ml})
\]

(8)

\[
\begin{align*}
\bar{s}_1 &= 0.315328(\%, v/v); \quad \bar{s}_2 = 0.657231(\%, w/v); \\
\bar{s}_3 &= 3.59959(\%, v/v); \quad \bar{s}_4 = 16.63918(\text{h}); \\
\bar{s}_y &= 3.648155(\text{U/ml})
\end{align*}
\]

(9)

The best performing soft-sensor expression (Eq. (6)) was obtained by choosing a specific set of operators containing only the four basic arithmetic operators, namely addition, subtraction, multiplication and division.

The optimality of the seven parameters in the GP-based soft-sensor model (Eq. (6)) was tested by subjecting it to the Levenberg–Marquardt (LM) nonlinear regression method [39]. Here, the specific objective was to explore whether there exists another set of parameter values that would improve the prediction...
accuracy and generalization performance of the soft-sensor Eq. (6). In this exercise, the LM method indeed found another set of values for the seven parameters appearing in Eq. (6); however, these LM-optimized parameters led to no significant improvement in the prediction accuracy and generalization performance of the resultant soft-sensor model.

The prediction accuracy and generalization performance of the GP-based soft-sensor were compared with those of the corresponding MLP and SVR based soft-sensors. The MLP-based soft-sensor was developed using the IBM SPSS (version 19) [51] software package. The training and test sets used in the development of the MLP-based soft-sensor were same as used in the GP simulations. Details of the optimal MLP-based soft-sensor are as follows: (i) training algorithm used: Error-back-propagation [25], (ii) transfer function for the hidden layer nodes: logistic sigmoid, (iii) transfer function for single output node: logistic sigmoid, (iv) number of hidden layers: 1, (v) number of hidden nodes: 3, (vi) momentum coefficient ($\mu$): 0.5, and (vii) initial learning rate ($\eta$): 0.15.

The SVR based soft-sensor was developed using the $\varepsilon$-SVR module of the Rapid Miner (Version 5.3.008) package [52]. The $\varepsilon$-SVR algorithm has two principal parameters, namely $C$ and $\varepsilon$, representing the regularization constant and radius of the tube, respectively. These were varied extensively to obtain an optimal SVR based soft-sensor model. The details of the optimal SVR-based model are: (i) kernel function used: radial basis function, (ii) $C = 3$, and (iii) $\varepsilon = 0.09$.

The magnitudes of $R$, $R^2$ and RMSE pertaining to the GP, MLP and SVR based soft-sensors are listed in Table 1. This table shows that the training and test set $R$ magnitudes of 0.965 and 0.964, respectively, pertaining to the GP-based soft-sensor are higher than the respective $R$ magnitudes of 0.946 and 0.937 for the MLP-based soft-sensor, and 0.930 and 0.914 for the SVR-based soft-sensor. It is also noticed that the training and test set RMSE magnitudes of 0.961 and 0.963 in respect of the lipase activity ($y_L$) predictions by the GP-based soft-sensor are significantly lower than the corresponding RMSE values of 1.426 and 1.542 for the MLP-based soft-sensor, and 1.432 and 1.461, for the SVR-based soft-sensor.

For comparing the goodness of the predictions made by the GP, MLP and SVR based soft-sensor models, a statistical test known as Hotelling t-test [53] was performed. It tests the null hypothesis ($H_0$) that $R_{AB} = R_{AC}$, where $R_{AB}$ ($R_{AC}$) refers to the correlation coefficient comparing the experimental and model-B (model-C) predicted $y_L$ (U/ml) values. The results of the Hotelling t-test are listed in Table 2. These indicate uniform rejection of the null hypothesis (at 95% confidence limit) comparing the correlation coefficient ($R$) magnitudes pertaining to the training and test set output ($y_L$) predictions by the model pairs GP–MLP, MLP–SVR and GP–SVR. It can thus be concluded that the differences in the $R$ magnitudes of the stated model pairs are statistically significant. From the $R^2$ and RMSE magnitudes listed in Table 1, it is noticed that the

![Fig. 7. A schematic of support vector regression using $\varepsilon$-insensitive loss function.](image)

**Table 1**

Prediction accuracies and the generalization performance of GP, MLP and SVR soft-sensor models for estimation of $y_L$ (U/ml).

<table>
<thead>
<tr>
<th>Model</th>
<th>Training set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$R$</td>
<td>$R^2$</td>
</tr>
<tr>
<td>GP</td>
<td>0.965</td>
<td>0.931</td>
</tr>
<tr>
<td>MLP</td>
<td>0.946</td>
<td>0.893</td>
</tr>
<tr>
<td>SVR</td>
<td>0.930</td>
<td>0.865</td>
</tr>
</tbody>
</table>
GP-based soft-sensor has outperformed both the MLP and SVR-based soft-sensors in accurately predicting and generalizing the lipase activity magnitudes. The parity plots of the experimental $y_1$ values and those predicted by the GP, MLP and SVR based soft-sensors are presented in Fig. 8. A comparison of these plots indicates that the scatter in the $y_1$ predictions made by the GP-based soft-sensor is in general low and in the region of lipase activity magnitudes of $<10$, the scatter is markedly lower when compared with the predictions of the MLP and SVR-based soft-sensors.

5.2. Case study-II: soft-sensor for estimating the accumulation of poly(3-hydroxybutyrate-co-3-hydroxyvalerate)

Bacterial polyhydroxyalkanoates (PHAs) are of interest as a raw material in the manufacture of biodegradable plastics. Among 150 PHAs identified so far, the occurrence of homopolymer of hydroxybutyrate (i.e., PHB) is widespread in different taxonomic groups of prokaryotes including cyanobacteria [54,55]. However, this homopolymer has poor physical properties, such as brittleness and low mechanical strength and thus its applications are limited. Also, its high melting temperature ($\approx 170^\circ C$) makes its processing a difficult task. On the other hand, poly(3-hydroxybutyrate-co-3-hydroxyvalerate), i.e., P(3HB-co-3HV) co-polymer, is less stiff and tougher. Incorporation of 3-hydroxyvalerate units not only reduces the crystallinity and melting temperature of the PHB homopolymer, but also imparts a better tensile strength to it.

Commercially, P(3HB-co-3HV) copolymer is produced using Wautersia eutropha bacterium (formerly known as Ralstonia eutropha) under fermentative conditions. However, the high carbon requirement and oxygen demand during fermentative production of the P(3HB-co-3HV) co-polymer makes its commercial manufacture uneconomical [56]. In this context, owing to their minimal nutrient requirement and photoautotrophic nature, the cyanobacteria can be considered as an alternative host system. Existing literature reveals that although a number of cyanobacterial species possess the ability to accumulate the homopolymer of PHB under photoautotrophic conditions, the Anabaena cylindrica 10C happens to be the only cyanobacterium that accumulates the P(3HB-co-3HV) under propionate-supplemented conditions with a maximum yield of 2% of the dry cell weight (dcw) [57]. A study conducted by Mallick et al. [58] showed the synthesis of P(3HB-co-3HV) co-polymer by an N$_2$-fixing cyanobacterium, Nostoc muscorum, under propionate supplemented conditions. For determining the accumulation of bacterial PHAs, the gas chromatography (GC) analysis of the PHA polymer is performed [58,59]. This analysis is time-consuming, tedious and requires a high level of expertise. Thus, availability of a soft-sensor is expected to significantly assist in the speedy estimation of the accumulation of PHAs and thereby in monitoring and controlling the bioprocess effectively.

Mallick et al. [58] produced P(3HB-co-3HV) co-polymer using Nostoc muscorum and optimized the process by rigorously examining the effects of various process variables and parameters, viz. concentrations of acetate and propionate, incubation period, and pH, on the process performance. The process data collected thereby (listed in Supplementary Table 2) have been used in the development of soft-sensor models predicting the accumulation of PHAs.

5.2.1. Results and discussion

The input space of the soft-sensor model consists of four process variables namely acetate concentration (% w/v) ($x_1$), propionate concentration (% w/v) ($x_2$), incubation period (days) ($x_3$) and pH ($x_4$), the model’s output variable is accumulated PHA ($y_P$) (% dwc). The example data set comprising process data from 30 experiments (see Supplementary Table 2) was randomly partitioned by the Eureqa software into 21 patterns (training set) and 9 patterns (test set); in Supplementary Table 2, the test data are marked with “***” symbol. As in the case study-I, the set of four basic arithmetic operators (addition, subtraction, multiplication and division) was used in the generation of candidate expressions. The GP-based overall optimal soft-sensor model displaying an excellent $y_P$ prediction accuracy and generalization performance is given as:

$$y_P = 1.0515 + 0.1015\hat{x}_2 - (0.06651\hat{x}_3/\hat{x}_1) - 0.803\hat{x}_3 - 0.09388\hat{x}_2 - 0.29563\hat{x}_3^2 - 0.37383\hat{x}_2^2 - 0.37383\hat{x}_3$$

(10)

where $\hat{y}_P$, $\hat{x}_1$, $\hat{x}_2$, $\hat{x}_3$, and $\hat{x}_4$, respectively, represent normalized values of the accumulated PHA ($y_P$), acetate concentration ($x_1$), propionate concentration ($x_2$), incubation period ($x_3$) and pH ($x_4$). The mean and standard deviation values used in the normalization procedure conducted using Eq. (7) are:

$$\hat{x}_1 = 0.108867, \quad \hat{x}_2 = 0.061, \quad \hat{x}_3 = 18 \text{ (days)}; \quad \hat{x}_4 = 8$$

$$\bar{y}_P = 17.16333 \text{ (} % \text{ dwc)}$$

(11)
s_1 = 0.078553 (%, w/v); \ s_2 = 0.035849 (%, w/v); \ s_3 = 3.638871 (days); \ s_4 = 0.909718; \ s_y = 10.69877 (\% dcw) (12)

It is seen that the soft-sensor model (Eq. (10)) contains eight parameters. Similar to the case study-I, the optimality of these parameters was tested using the LM nonlinear regression algorithm [39]. However, in this case study too, the LM-based parameter optimization did not lead to any significant improvement in the prediction accuracy and generalization capability of the resultant soft-sensor model. Thus, in both the case studies, the GP-searched parameters of the respective soft-sensor models proved to be optimal.

The performance of the GP-based soft-sensor was compared with that of the MLP neural network and SVR based soft-sensor models. These models used the same training and test data sets utilized in the GP-based soft-sensor development. An optimal MLP-based soft-sensor was developed using the IBM SPSS (Version 19) package and its details are as follows: (i) MLP training algorithm used: EBP [25], (ii) transfer function for hidden layer nodes: hyperbolic tangent, (iii) transfer function for the output layer node: hyperbolic tangent, (iv) number of hidden layers: 1, (v) number of hidden nodes: 3, (vi) momentum coefficient ($\mu$): 0.004, and (vii) initial learning rate ($\eta$): 0.05.

An SVR-based optimal soft-sensor model for predicting the accumulation of PHA ($y_p$) was developed using the Rapid Miner package. The details of the optimal SVR-based soft-sensor are as follows: (i) kernel function used: radial basis function; (ii) C (regularization constant) = 2; and (iii) $\varepsilon$ (width of the tube) = 0.05.

Table 3 provides a comparison of the prediction accuracy and generalization capability of the GP, MLP and SVR based soft-sensors predicting accumulation PHA ($y_p$). It is seen from the table that the accumulated PHA ($y_p$) predictions by the GP based soft-sensor model have resulted in higher $R$ magnitudes of 0.982 and 0.979 for the training and test set data, respectively than the corresponding $R$ magnitudes of 0.938 and 0.938, for the MLP based soft-sensor, and 0.978 and 0.969 for the SVR-based soft-sensor. It is also observed that the $y_p$ predictions by the GP-based soft-sensor have resulted in lower RMSE magnitudes for the training and test set data than the corresponding RMSE magnitudes in respect of the MLP and SVR-based soft-sensors. As can be noticed from Table 3, the differences between the $R$ magnitudes of the GP-based soft-sensor and the corresponding values pertaining to the SVR based soft-sensor are marginal.

The results of the Hotelling $t$-test performed for comparing the $R$ magnitudes in respect of the three soft-sensor pairs, namely GP-MLP, MLP-SVR and GP-SVR are given in Table 4. These results indicate that barring the difference between the training set $R$ values of 0.982 and 0.938 pertaining to the GP-MLP soft-sensor pair, all other differences in the $R$ values are statistically insignificant at 95% confidence limit. The RMSE values however indicate that the GP model performs significantly better than the MLP-based model and only marginally better than the SVR based one. The parity plots of the experimental values of the accumulated PHA and those predicted by the GP, MLP and SVR based soft-sensors are displayed in Fig. 9. A comparison of these plots indicates that the $y_p$ predictions by the GP-based soft-sensor exhibit least scatter when compared with the predictions made by the MLP and SVR based soft-sensor models. Overall, the above-stated comparison of the GP, SVR and MLP based soft-sensors indicates that while the first two exhibit comparable prediction and generalization capabilities, the performance of the third soft-sensor model is inferior.

### Table 3
Prediction accuracies and the generalization performance of GP, MLP, and SVR-based soft-sensor models for the estimation of $y_p$ (\% dcw).

<table>
<thead>
<tr>
<th>Model</th>
<th>Training set</th>
<th>Test set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$R$</td>
<td>$R^2$</td>
<td>RMSE</td>
</tr>
<tr>
<td>GP</td>
<td>0.982</td>
<td>0.964</td>
<td>2.067</td>
</tr>
<tr>
<td>MLP</td>
<td>0.938</td>
<td>0.879</td>
<td>4.058</td>
</tr>
<tr>
<td>SVR</td>
<td>0.978</td>
<td>0.956</td>
<td>2.255</td>
</tr>
</tbody>
</table>

### Table 4
Results of Hotelling $t$-test for comparing correlation coefficient ($R$) values of a pair of soft-sensor models in respect of case study-II.

<table>
<thead>
<tr>
<th>Model pair (B-C)</th>
<th>Training set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$df$</td>
<td>$t$</td>
</tr>
<tr>
<td>GP-MLP</td>
<td>18</td>
<td>2.766</td>
</tr>
<tr>
<td>MLP-SVR</td>
<td>18</td>
<td>0.338</td>
</tr>
<tr>
<td>SVR-GP</td>
<td>18</td>
<td>-2.521</td>
</tr>
</tbody>
</table>

$^*$ $H_0$: $R_{cal}=R_{exp}$; where $A$ denotes experimental values of $y_p$; $t_{critical}$ ($\alpha=0.05$; degrees of freedom, $df=18$): 1.7341; $t_{critical}$ ($\alpha=0.05$; $df=6$): 1.9432; reject $H_0$ if $|t|>|t_{critical}|$.

6. Conclusion

In the present paper, a novel and exclusively data-driven AI-based modeling formalism, namely genetic programming has been introduced for the development of soft-sensors for the bioprocesses. The efficacy of the GP-based soft-sensors has been successfully demonstrated by conducting two case studies
involving production of the lipase enzyme and bacterial poly (3-hydroxybutyrate-co-3-hydroxyvalerate) copolymer, respectively. In these case studies, the soft-sensors predict the time-dependent lipase activity and the amount of accumulated polyhydroxalkanoates, respectively. The prediction and generalization performance of the GP-based soft-sensors was compared with those developed using the MLP neural network and SVR based soft-sensors. Results obtained from the case study-I indicate the superior prediction accuracy and generalization capability of the GP-based soft-sensor vis-a-vis MLP and SVR based soft-sensors. In case study-II, the GP-based soft-sensor has performed significantly (marginally) better than the MLP (SVR) based soft-sensor. These results clearly indicate that for processes exhibiting non-linear behavior, no single data-driven modeling formalism can consistently outperform the other non-linear modeling methods. Accordingly, the principal aim of this study is to introduce the feature-rich and attractive, yet seldom utilized artificial intelligence based GP formalism, as a promising candidate for the development of soft-sensors for bioprocesses. It may be noted that the GP-based soft-sensor development strategy exemplified here is not restricted to only bioprocesses but has a wider application potential in developing soft-sensor models for various other processes including petrochemical, petroleum, polymer and pharmaceutical processes.

**Appendix A.**

Formula (expression) building blocks available in *Eureka Formulize* (version 0.98.01) [17] software package

- Basic: constant, integer constant, input variable, addition, subtraction, multiplication, division, negation
- Trigonometry: sine, cosine, tangent
- Exponential: exponential, natural logarithmic, factorial, power, square root
- Squashing: logistic function, step function, sign function, gaussian function, hyperbolic function, error function, complementary function
- History: delayed variable, simple moving average, weighted moving average, modified moving average, simple moving median
- Logical: if-then-else, equal-to, less-than-or-equal, greater-than, greater-than-or-equal, AND, OR, XOR, NOT
- Other: minimum, maximum, modulo, floor, ceiling, round, absolute value, hyperbolic sine, hyperbolic cosine
- Inverse trigonometry: arcsine, arccosine, arctangent, two-argument arctangent, inverse hyperbolic sine, inverse hyperbolic cosine, inverse hyperbolic tangent

**Appendix B. Supplementary data**

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jhe.2014.02.007.

**References**

26. S.S. Tambe, B.D. Kulkarni, P.B. Deshpande, Elements of Artificial Neural Networks with Selected Applications in Chemical Engineering, and Chemical & Biological Sciences, Simulations & Advanced Controls, Louisville, 1996.