Korean J. Chem. Eng., **26**(5), 1175-1185 (2009)
DOI: 10.2478/s11814-009-0195-6

Support vector regression with parameter tuning assisted by differential evolution technique: Study on pressure drop of slurry flow in pipeline

Sandip Kumar Lahiri† and Kartik Chandra Ghanta

Department of Chemical Engineering, NIT, Durgapur, West Bengal, India
(*Received 18 April 2008 • accepted 1 February 2009*)

Abstract−This paper describes a robust support vector regression (SVR) methodology that offers superior performance for important process engineering problems. The method incorporates hybrid support vector regression and differential evolution technique (SVR-DE) for efficient tuning of SVR meta parameters. The algorithm has been applied for prediction of pressure drop of solid liquid slurry flow. A comparison with selected correlations in the literature showed that the developed SVR correlation noticeably improved prediction of pressure drop over a wide range of operating conditions, physical properties, and pipe diameters.

Key words: Support Vector Regression, Differential Evolution, Slurry Pressure Drop

INTRODUCTION

Conventionally, two approaches, namely *phenomenological* (first principles) and *empirical*, are employed for slurry flow modeling. In phenomenological modeling, a detailed knowledge of the solid liquid interaction and associated heat, momentum and mass transport phenomena is required to represent mass, momentum, and energy balances. The advantages of a phenomenological model are: (i) since it represents physico-chemical phenomenon underlying the process explicitly, it provides a valuable insight into the process behavior, and (ii) it possesses extrapolation ability.

Owing to the complex nature of many multiphase phase slurry processes, the underlying physico-chemical phenomenon is seldom fully understood. Also, collection of the requisite phenomenological information is costly, time-consuming and tedious, and therefore development of phenomenological process models poses considerable practical difficulties. Moreover, nonlinear behavior being common in multiphase slurry processes, it leads to complex nonlinear models, which in most cases are not amenable to analytical solutions and thus require computationally intensive numerical methods for obtaining solutions. The difficulty associated with the construction and solution of phenomenological models necessitates exploration of alternative modeling formalisms. Modeling using empirical (regression) methods is one such alternative. In conventional empirical modeling, appropriate linear or nonlinear models are constructed exclusively from the process input-output data without invoking the process phenomenology. A fundamental deficiency of the conventional empirical modeling approach is that the structure (functional form) of the data-fitting model must be specified a priori. Satisfying this requirement, especially for nonlinearly behaving processes, is cumbersome since it involves selecting heuristically an appropriate nonlinear model structure from numerous alternatives.

In the last decade, artificial neural networks (ANNs) and more recently support vector regression (SVR) have emerged as two attractive tools for nonlinear modeling, especially in situations where the development of phenomenological or conventional regression models becomes impractical or cumbersome. In recent years, support vector regression (SVR) [28-30], which is a statistical learning theory based machine learning formalism, has gained popularity over ANN due to its many attractive features and promising empirical performance. The salient features of SVR are: (i) like ANNs, SVR is an exclusively data based nonlinear modeling paradigm; (ii) SVR based models are based on the principle of structural risk minimization, which equips them with greater potential to generalize; (iii) parameters of an SVR model are obtained by solving a quadratic optimization problem; (iv) the objective function in SVR being of quadratic form, it possesses a single minimum, thus avoiding the heuristic procedure involved in locating the global or the deepest local minimum on the error surface; and (v) in SVR, the inputs are first nonlinearly mapped into a high dimensional feature space wherein they are correlated linearly with the output; (vi) support vector machine (SVM) always has solution, which can be quickly obtained by a standard algorithm (quadratic programming); (vii) SVM need not determine network topology in advance, which can be automatically obtained when training process ends; and (viii) SVM can be regarded as a representation of information reducing (dimension reducing). It can solve high-dimension problems and therefore avoid the "curse of dimensionality."

This study is motivated by a growing popularity of support vector machines (SVM) for regression problems. By virtue of their many desirable learning properties, SVMs have been widely employed in many different fields [Burbidge et al., 2001; 16; Bao and Sun, 2002; Guo, Li and Chan, 2001; Suykens, Vandewalle and Moore, 2001]. Although the foundation of the SVR paradigm was laid down in the mid 1990s, its chemical engineering applications such as fault detection [Agarwal., 2003; Jack et al., 2002] have emerged only recently.

However, many SVM regression application studies are performed by 'expert' users having good understanding of SVM methodology. Since the quality of SVM models depends on a proper setting of SVM meta-parameters, the main issue for practitioners trying to

†To whom correspondence should be addressed.
E-mail: sk_lahiri@hotmail.com

apply SVM regression is how to set these parameter values (to ensure good generalization performance) for a given data set. Whereas existing sources on SVM regression give some recommendations on appropriate setting of SVM parameters, there is clearly no consensus and (plenty of) contradictory opinions.

Most of the earlier approaches use trial and error procedures for tuning the SVM parameters while trying to minimize the training and test errors. Such an approach apart from consuming enormous time may not really obtain the best possible performance.

Conventionally, various deterministic gradient based methods [8] are used for optimizing a process model. Most of these methods, however, require that the objective function should be smooth, continuous, and differentiable. The SVR models cannot be guaranteed to be smooth, especially in regions wherein the input-output data (training set) used in model building is located sparsely. Hence, gradient-based methods cannot be used efficiently for optimizing the input space of an SVR model. In such situations, an efficient optimization formalism known as differential evolution (DEs), which is lenient towards the form of the objective function, can be used. In the recent years, DEs that are members of the stochastic optimization formalisms have been used with a great success in solving problems involving very large search spaces. The DEs were originally developed as genetic engineering models mimicking population evolution in natural systems. Specifically, DE, like genetic algorithm (GA), enforce the "survival-of the- fittest" and "genetic propagation of characteristics" principles of biological evolution for searching the solution space of an optimization problem. DE has been used to design several complex digital filters [20] and to design fuzzy logic controllers [23]. DE can also be used for parameter estimations; e.g., Babu and Sastry, 1999 used DE for the estimation of effective heat transfer parameters in trickle-bed reactors using radial temperature profile measurements. They concluded that DE takes less computational time to converge compared to the existing techniques without compromising with the accuracy of the parameter estimates.

In the present paper, SVR formalism is integrated with differential evolution to arrive at modeling and optimization strategies. The strategy, henceforth referred to as "SVR-DE," uses an SVR as the nonlinear process modeling paradigm, and the DE for optimizing the meta-parameters of the SVR model such that an improved prediction performance is realized. To our knowledge, a hybrid involving SVR and DE is being used for the first time for chemical process modeling and optimization. In the present work, we propose a hybrid support vector regression-differential evolution (SVR-DE) approach for tuning the SVR meta parameters and illustrate it by applying it for predicting the pressure drop of solid liquid flow.

The paper is organized as follows: section 2 describes process modeling using SVR methods; a brief introduction of DE is given in section 3. The optimization of the SVR model using DEs is explained in section 4. Usage of SVR-DE strategies for case studies of pressure drop in slurry pipeline along with the results is described in sections 5 and 6, respectively. Finally, section 7 gives a summary of the study.

SUPPORT VECTOR REGRESSION (SVR) MODELING

Support vector regression (SVR) is an adaptation of a recent statistical learning theory based classification paradigm, namely support vector machines [28]. The SVR formulation follows the structural risk minimization (SRM) principle, as opposed to the empirical risk minimization (ERM) approach, which is commonly employed within statistical machine learning methods and also in training ANNs. In SRM, an upper bound on the generalization error is minimized as opposed to the ERM, which minimizes the prediction error on the training data. This equips the SVR with a greater potential to generalize the input-output relationship learned during its training phase for making good predictions for new input data. The SVR is a linear method in a high dimensional feature space, which is nonlinearly related to the input space. Though the linear algorithm works in the high dimensional feature space, in practice it does not involve any computations in that space, since through the usage of kernels all necessary computations are performed directly in the input space. In the following, the basic concepts of SVR are introduced. A more detailed description of SVR can be found in [3,25,26,29].

1. Support Vector Regression: At a Glance

Consider a training data set $g=\{(x_1, y_1), (x_2, y_2), \ldots, (x_P, y_P)\}$, such that $xi \in \upsilon^N$ is a vector of input variables and $y_i \in \upsilon$ is the corresponding scalar output (target) value. Here, the modeling objective is to find a regression function, y=f(x), such that it accurately predicts the outputs {y} corresponding to a new set of input-output examples, {(x, y)}, which are drawn from the same underlying joint probability distribution, P(x, y), as the training set. To fulfill the stated goal, SVR considers following linear estimation function.

$$f(x)=<w, \phi(x)>+b \tag{1}$$

where w denotes the weight vector; b refers to a constant known as "bias"; f(x) denotes a function termed *feature*, and $<w, \phi(x)>$ represents the dot product in the feature space, *l*, such ϕ: $x \rightarrow l$, $w \in l$. In SVR, the input data vector, x, is mapped into a high dimensional feature space, *l*, via a nonlinear mapping function, ϕ, and a linear regression is performed in this space for predicting y. Thus, the problem of nonlinear regression in lower dimensional input space υ^N is transformed into a linear regression in the high dimensional feature space, *l*. Accordingly, the original optimization problem involving nonlinear regression is transformed into finding the flattest function in the feature space *l* and not in the input space, x. The unknown parameters w and b in Eq. (4) are estimated by using the training set, g. To avoid over-fitting and thereby improving the generalization capability, the following regularized functional involving summation of the empirical risk and a complexity term w^2, is minimized:

$$R_{reg}[f] = R_{emp}[f] + \lambda|w|^2 = \sum_{i=1}^{P} C(f(xi) - yi) + \lambda\|w\|^2 \tag{2}$$

where Rreg and Remp denote the regression and empirical risks, respectively; w^2 is the Euclidean norm; C(.) is a cost function measuring the empirical risk, and $\lambda>0$ is a regularization constant. For a given function, f, the regression risk (test set error), Rreg(f), is the possible error committed by the function f in predicting the output corresponding to a new (test) example input vector drawn randomly from the same sample probability distribution, P(x, y), as the training set. The empirical risk Remp (f), represents the error (termed "training set error") committed in predicting the outputs of the training set inputs. Minimization task described in Eq. (2) involves: (i) minimization of the empirical loss function Remp(f) and, (ii) ob-

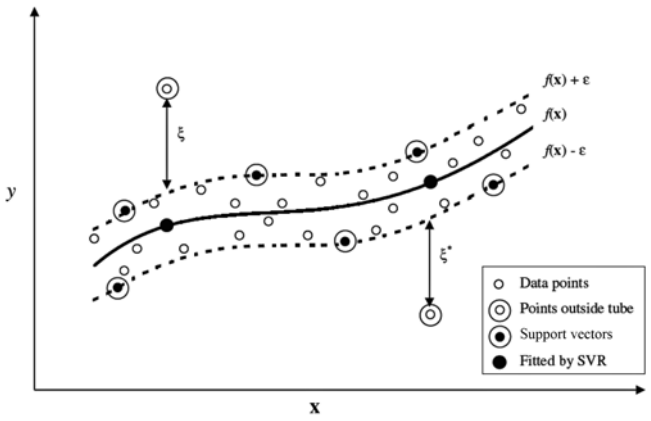

Fig. 1. A schematic diagram of support vector regression using e-sensitive loss function.

taining as small a w as possible, using the training set g. The commonly used loss function is the "e-insensitive loss function" given as [30]:

$$C(f(x)-y)=|f(x)-y|-\varepsilon \quad \text{For } |f(x)-y|\geq\varepsilon$$
$$=0 \quad \text{otherwise} \tag{3}$$

where ε is a precision parameter representing the radius of the tube located around the regression function (see Fig. 1); the region enclosed by the tube is known as "e-intensive zone." The SVR algorithm attempts to position the tube around the data as shown in Fig. 1.

The optimization criterion in Eq. (3) penalizes those data points whose y values lie more than ε distance away from the fitted function, f(x). In Fig. 1, the size of the stated excess positive and negative deviations is depicted by ζ and ζ*, which are termed "slack" variables. Outside of the $[-\varepsilon, \varepsilon]$ region, the slack variables assume non-zero values. The SVR fits f(x) to the data in a manner such that: (i) the training error is minimized by minimizing ζ and ζ* and, (ii) w^2 is minimized to increase the flatness of f(x) or to penalize over complexity of the fitting function. Vapnik 1998 showed that the following function possessing a finite number of parameters can minimize the regularized function in Eq. (2).

$$f(x,\alpha,\alpha^*)=\sum_{i=1}^{P}(\alpha i-\alpha i^*)K(x,xi)+b \tag{4}$$

where, αi and αi* (>=0) are the coefficients (Lagrange multipliers) satisfying $\alpha i\ \alpha i^*=0$, i=1, 2, …, P, and K(x, xi) denotes the so called 'kernel' function describing the dot product in the feature space. The kernel function is defined in terms of the dot product of the mapping function as given by

$$K(xi, xj)=<\phi(xi), \phi(xj)> \tag{5}$$

The advantage of this formulation (Eqs. (4) and (5)) is that for many choices of the set $\{\phi i(x)\}$, including infinite dimensional sets, the form of K is analytically known and very simple. Accordingly, the dot product in the feature space i *l* can be computed without actually mapping the vectors xi and xj into that space (i.e., computation of ϕ(x(i) and ϕ(x(j)). There exist several choices for the kernel function K (Refer Table 1); the two commonly used kernel functions, namely, radial basis function (RBF) and *n*th degree polynomial, are defined below in Eqs. (6) and (7), respectively.

$$K(xi,xj)=\exp\left(\frac{\|xi-xj\|^2}{2\sigma^2}\right) \tag{6}$$

$$K(xi,xj)=[1+(xi,xj)^n] \tag{7}$$

In Eq. (4), the coefficients αi and αi* are obtained by solving following quadratic programming problem.

Maximize:

$$R(\alpha^*,\alpha)=-0.5\sum_{i,j=1}^{P}(\alpha i^*-\alpha i)(\alpha j^*-\alpha j)K(xi,xj)$$
$$-\varepsilon\sum_{i=1}^{P}(\alpha i^*+\alpha i)+\sum_{i=1}^{P}yi(\alpha i^*-\alpha i)$$
$$\text{subject to constraints: } 0\leq\alpha i,\alpha i^*\leq C,\forall i \text{ and } \sum_{i=1}^{P}(\alpha i^*-\alpha i)=0 \tag{8}$$

Having estimated α, α* and b, using a suitable quadratic programming algorithm, the SVR-based regression function takes the form

$$f(x,w)=f(x,\alpha,\alpha^*)=\sum_{i=1}^{P}(\alpha i^*-\alpha i)K(xi,x)+b \tag{9}$$

where, vector w is described in terms of the Lagrange multipliers α and α*. Owing to the specific character of the above-described quadratic programming problem, only some of the coefficients, ($\alpha i^*-\alpha i$) are non-zero and the corresponding input vectors, xi, are called support vectors (SVs). The SVs can be thought of as the most informative data points that compress the information content of the

Table 1. Different kernel type

Case	Name of Kernel	Equation
Case 1	Linear	k=u*v′;
Case 2	Polynomial	k=(u*v′+1)^p1; p1 is degree of polynomial
Case 3	Gaussian radial basis function	k=exp(−(u−v)*(u−v)′/(2*p1^2)); p1 is width of rbfs (sigma)
Case 4	Exponential radial basis function	k=exp(−sqrt((u−v)*(u−v)′)/(2*p1^2)); p1 is width of rbfs (sigma)
Case 5	Splines	z=1+u.*v+(1/2)*u.*v.*min(u, v)−(1/6)*(min(u, v)).^3; k=prod(z);
Case 6	B splines	z=0; for r=0: 2*(p1+1) z=z+(−1)^r*binomial(2*(p1+1), r)*(max(0, u−v+p1+1−r)).^(2*p1+1); end k=prod(z); p1 is degree of bspline

Where u, v-kernel arguments

training set. The coefficients α and α^* have an intuitive interpretation as forces pushing and pulling the regression estimate f(xi) towards the measurements, yi. In Eq. (9), the bias parameter, b, can be computed as follows:

$$b=\begin{cases} yi-f(xi)_{b=0}-\varepsilon & \text{For } \alpha i\in(0,C) \\ yi-f(xi)_{b=0}+\varepsilon & \text{For } \alpha i^*\in(0,C) \end{cases}$$

where, **x**i and yi denote the *i*th support vector and the corresponding target output, respectively. In the SVR formulation, C and ε are two user-specified free parameters; while C represents the trade-off between the model-complexity and the approximation error, ε signifies the width of the ε-insensitive zone used to fit the training data. The stated free parameters together with the specific form of the kernel function control the accuracy and generalization performance of the regression estimate. The procedure of judicious selection of C and ε is explained by [6].

2. Performance Estimation of SVR

It is well known that SVM generalization performance (estimation accuracy) depends on a good setting of meta-parameters parameters C, ε, and kernel parameters such as kernel type, loss function type and the kernel parameters. The problem of optimal parameter selection is further complicated by the fact that SVM model complexity, and hence its generalization performance, depends on all five parameters.

Selecting a particular kernel type and kernel function parameters is usually based on application-domain knowledge and also should reflect distribution of input (**x**) values of the training data. Parameter C determines the trade-off between the model complexity (flatness) and the degree to which deviations larger than ε are tolerated in optimization formulation. For example, if C is too large (infinity), then the objective is to minimize the empirical risk only, without regard to model complexity part in the optimization formulation.

Parameter ε controls the width of the ε-insensitive zone, used to fit the training data [5,30].

The value of ε can affect the number of support vectors used to construct the regression function. The bigger ε, the fewer support vectors are selected. On the other hand, bigger ε-values result in more 'flat' estimates. Hence, both C and ε-values affect model complexity (but in a different way).

To minimize the generalization error, these parameters should be properly optimized.

Existing practical approaches to the choice of C and ε can be summarized as follows (Cherkassy and Ma):

• Parameters C and ε are selected by users based on a priori knowledge and/or user expertise (Cherkassky and Mulier, 1998; Vapnik, 1999; Vapnik, 1998; B. Scholkopf, Burges, Smola, 1999). Obviously, this approach is not appropriate for non-expert users. Based on the observation that support vectors lie outside the ε-tube and the SVM model complexity strongly depends on the number of support vectors, Schölkopf, Bartlett, Smola, and Williamson, 1998 suggest to control another parameter ν (the fraction of points outside the ε-tube) instead of ε. Under this approach, parameter ν has to be user-defined. Similarly, Mattera and Haykin, 1999 propose to choose the ε-value so that the percentage of support vectors in the SVM regression model is around 50% of the number of samples. However, one can easily show examples when optimal generalization performance is achieved with the number of support vectors larger or smaller than 50%.

• Smola, Murata, Schölkopf and K. Muller, 1998 and Kwok, 2001 proposed asymptotically optimal ε-values proportional to noise variance, in agreement with general sources on SVM (Cherkassky and Mulier, 1998; Vapnik, 1999; Vapnik, 1998). The main practical drawback of such proposals is that they do not reflect sample size. Intuitively, the value of ε should be smaller for a larger sample size than for a small sample size (with the same level of noise).

• *Selection of* ε. It is well-known that the value of should be proportional to the input noise level, that is, $\varepsilon \propto \sigma$ (Cherkassky and Mulier, 1998; Vapnik, 1999; Vapnik, 1998; Kwok, 2001; A. Smola, N. Murata, B. Schölkopf and K. Muller, 1998).

Vladimir Cherkassky and Yunqian Ma propose the following (empirical) dependency:

$$\varepsilon=\tau\sigma\sqrt{\frac{\ln n}{n}}$$

Based on empirical tuning, they propose that a constant value τ=3 gives good performance for various data set sizes, noise levels and target functions for SVM regression. Here, n is number of training data sample. They assume that the standard deviation of noise σ is known or can be estimated from data which is again a difficult task for non expert user.

• Selecting parameter C equal to the range of output values (Mattera and Haykin, 1999). This is a reasonable proposal, but it does not take into account the possible effect of outliers in the training data.

Vladimir Cherkassky and Yunqian Ma propose to use instead the following prescription for regularization parameter:

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

where $\bar{y}$ is the mean of the training responses (outputs), and σ_y is the standard deviation of the training response values. They claim this prescription can effectively handle outliers in the training data. However, the proposed value of C-parameter is derived and applicable for RBF kernels only.

• Using cross-validation for parameter choice (Cherkassky and Mulier, 1998; Scholkopf, Burges, Smola, 1999). This is very computation- and data-intensive.

• Several recent references present statistical account of SVM regression (Smola and Schölkopf, 1998; Hastie, Tibshirani and Friedman, 2001) where the ε-parameter is associated with the choice of the loss function (and hence could be optimally tuned to particular noise density), whereas the C parameter is interpreted as a traditional regularization parameter in formulation that can be estimated for example by cross-validation (Hastie, Tibshirani and Friedman, 2001).

• Selecting a particular kernel type and kernel function parameters is usually based on application-domain knowledge and also should reflect distribution of input (**x**) values of the training data. Very little literature is available to throw light on this.

As evident from the above, there is no shortage of conflicting opinions on optimal setting of SVM regression parameters. Existing software implementations of SVM regression usually treat SVM meta-parameters as user-defined inputs. For non-expert users it is very difficult task to choose these parameters as they have no prior knowledge for these parameters for their data. In such a situation,

users normally rely on trial and error method. Such an approach apart from consuming enormous time may not really obtain the best possible performance. In this paper we present a hybrid SVR-DE approach, which not only relieves the user from choosing these meta parameters but also finds the optimum values of these parameters to minimize the generalization error.

DIFFERENTIAL EVOLUTION (DE): AT A GLANCE

Having developed an SVR-based process model, a DE algorithm is used to optimize the *N*-dimensional input space (**x**) of the SVR model. The DEs were originally developed as the genetic engineering models mimicking population evolution in natural systems. Specifically, DE, like genetic algorithm (GA), enforces the "survival-of the-fittest" and "genetic propagation of characteristics" principles of biological evolution for searching the solution space of an optimization problem. The principal features possessed by the DEs are: (i) they require only scalar values and not the second- and/or first-order derivatives of the objective function, (ii) capability to handle nonlinear and noisy objective functions, (iii) they perform global search and thus are more likely to arrive at or near the global optimum, and (iv) DEs do not impose pre-conditions, such as smoothness, differentiability and continuity, on the form of the objective function.

Differential evolution (DE), an improved version of GA, is an exceptionally simple evolution strategy that is significantly faster and more robust at numerical optimization and is more likely to find a function's true global optimum. Unlike simple GA that uses a binary coding for representing problem parameters, DE uses real coding of floating point numbers. The mutation operator here is addition instead of bit-wise flipping used in GA. And DE uses non-uniform crossover and tournament selection operators to create new solution strings. Among the Des advantages are simple structure, ease of use, speed and robustness. It can be used for optimizing functions with real variables and many local optima.

This paper demonstrates the successful application of differential evolution to a practical optimization problem. As already stated, DE in principle is similar to GA. So, as in GA, we use a population of points in our search for the optimum. The population size is denoted by NP. The dimension of each vector is denoted by D. The main operation is the NP number of competitions that are to be carried out to decide the next generation.

To start with, we have a population of NP vectors within the range of the objective function. We select one of these NP vectors as our *target vector.* We then randomly select two vectors from the population and find the difference between them (vector subtraction). This difference is multiplied by a factor F (specified at the start) and added to a third randomly selected vector. The result is called the *noisy random vector.* Subsequently, crossover is performed between the target vector and noisy random vector to produce the *trial vector.* Then, a competition between the trial vector and target vector is performed and the winner is replaced into the population. The same procedure is carried out NP times to decide the next generation of vectors. This sequence is continued till some convergence criterion is met. This summarizes the basic procedure carried out in differential evolution. The details of this procedure are described below.

1. Steps Performed in DE

Assume that the objective function is of D dimensions and that it has to be optimized. The weighting constants F and the crossover constant CR are specified.

Step 1 Generate NP random vectors as the initial population: generate (NP * D) random numbers and linearize the range between 0 and 1 to cover the entire range of the function. From these (NP * D) numbers, generate NP random vectors, each of dimension D, by mapping the random numbers over the range of the function.

Step 2 Choose a target vector from the population of size NP: first generate a random number between 0 and 1. From the value of the random number decide which population member is to be selected as the target vector (Xi) (a linear mapping rule can be used).

Step 3 Choose two vectors at random from the population and find the weighted difference: Generate two random numbers. Decide which two population members are to be selected (Xa, Xb). Find the vector difference between the two vectors (Xa−Xb). Multiply this difference by F to obtain the weighted difference.

Weighted difference=F (Xa−Xb)

Step 4 Find the noisy random vector: Generate a random number. Choose a third random vector from the population (Xc). Add this vector to the weighted difference to obtain the noisy random vector (X′c).

Step 5 Perform crossover between Xi and X′c to find Xt, the trial vector: Generate D random numbers. For each of the D dimensions, if the random number is greater than CR, copy the value from Xi into the trial vector; if the random number is less than CR, copy the value from X′c into the trial vector.

Step 6 Calculate the cost of the trial vector and the target vector: For a minimization problem, calculate the function value directly and this is the cost. For a maximization problem, transform the objective function f(x) using the rule F(x)=1/[1+f(x)] and calculate the value of the cost. Alternatively, directly calculate the value of f(x) and this yields the profit. In case cost is calculated, the vector that yields the lesser cost replaces the population member in the initial population. In case profit is calculated, the vector with the greater profit replaces the population member in the initial population.

Step 1-6 are continued until some stopping criterion is met. This may be of two kinds. One may be some convergence criterion that states that the error in the minimum or maximum between two previous generations should be less than some specified value. The other may be an upper bound on the number of generations. The stopping criterion may be a combination of the two. Either way, once the stopping criterion is met, the computations are terminated.

Choosing DE Key Parameters NP, F, and CR is seldom difficult and some general guidelines are available. Normally, NP ought to be about 5 to 10 times the number of parameters in a vector. As for F, it lies in the range 0.4 to 1.0. Initially, F=0.5 can be tried, then F and/or NP is increased if the population converges prematurely. A good first choice for CR is 0.1, but in general CR should be as large a possible (Price and Storn, 1997).

Already, DE has been successfully applied for solving several complex problems and is now being identified as a potential source for accurate and faster optimization.

DE-BASED OPTIMIZATION OF SVR MODELS

There are different measures by which SVM performance is as-

sessed, validation and leave-one-out error estimates being the most commonly used. Here we divide the total available data as training data (75% of data) and test data (25% data chosen randomly). Whereas, the SVR algorithm was trained on training data but the SVR performance is estimated on test data.

The statistical analysis of SVR prediction is based on the following performance criteria:

1. The average absolute relative error (AARE) on test data should be minimum:

$$\mathrm{AARE}=\frac{1}{\mathrm{N}}\sum_{1}^{N}\left|\left(\frac{\mathrm{ypredicted}-\mathrm{yexperiment}}{\mathrm{yexperimental}}\right)\right|$$

2. The standard deviation of error (σ) on test data should be minimum:

$$\sigma=\sqrt{\sum_{1}^{N}\frac{1}{\mathrm{N}-1/\mathrm{yexperimenatl}_{(i)}|-\mathrm{AARE}]^{2}}[|(\mathrm{ypredicted}_{(i)}-\mathrm{yexperimental}_{(i)})}$$

3. The cross-correlation co-efficient (R) between input and output should be around unity:

$$\mathrm{R}=\frac{\sum_{i=1}^{N}(\mathrm{yexperimental(i)}-\mathrm{yexperimental(mean)})(\mathrm{ypredicted(i)}-\mathrm{ypredicted(mean)})}{\sqrt{\sum_{i=1}^{N}(\mathrm{yexperimental(i)}-\mathrm{yexperimental(mean)})^{2}}\sqrt{\sum_{i=1}^{N}(\mathrm{ypredicted(i)}-\mathrm{ypredicted(mean)})^{2}}}$$

SVM learning is considered successful only if the system can perform well on test data on which the system has not been trained. The above five parameters of SVR are optimized by DE algorithm stated below.

The objective function and the optimal problem of SVR model of the present study are represented as

Minimize
AARE(**X**) on test set
X∈ {x1, x2, x3, x4, x5}
where
x1={0 to 10,000}
x2={0 to 1}
x3={1, 2}
x4={1, 2, ..., 6}
x5={1, 2, .., 6}

The objective function is minimization of average absolute relative error (AARE) on the test set and **X** is a solution string representing a design configuration. The design variable x1 takes any values for C in the range of 0.0 to 10,000. x2 represents the ε (epsilon) taking any values in the range of 0.0 to 1. x3 represents the loss function types: e-insensitive loss function and Huber loss function represented by the numbers 1 and 2, respectively. X4 represents the Kernel types: all 6 kernels given in Table 1 represented by the numbers 1 to 6 respectively. The variable x5 takes six values of the kernel parameters (represents degree of polynomials etc.) in the range 1 to 6 (1, 2, 3, 4, 5, 6) represented by numbers 1 to 6.

The total number of design combinations with these variables is 100×100×2×6×6=720,000. This means that if an exhaustive search is to be performed it will take at the maximum 720,000 function evaluations before arriving at the global minimum AARE for the test set (assuming 100 trials for each to arrive optimum C and ε). So the strategy which takes few function evaluations is the best one. Considering minimization of AARE as the objective function, differential evolution technique is applied to find the optimum design configuration of SVR model.

CASE STUDY: PREDICTION OF PRESSURE DROP IN SOLID LIQUID SLURRY FLOW

1. Background and Importance of Pressure Drop

Pipeline transport has been a progressive technology for conveying a large quantity of bulk materials. The modern way of pipelining prefers the concentrated slurries since hydraulic transport of dense hydro-mixtures can bring several advantages. Compared to a mechanical transport, the use of a pipeline ensures a dust-free environment, demands substantially less space, makes possible full automation and requires a minimum of operating staff. On the other hand, it brings higher operational pressures and considerable demands for a high quality of pumping equipment and control system.

Power consumption represents a substantial portion of the overall pipeline transport operational costs. For that reason great attention was paid to reduction of the hydraulic losses. The prediction of pressure drop of slurries and the understanding of rheological behavior makes it possible to optimize energy and water requirements.

Despite the large area of application, the available models describing the suspension mechanism do not completely satisfy engineering needs. The behavior of solids in liquid flowing through pipelines has been the subject of continuing investigation since the turn of the 19th century. From the literature [7, 10, 18], Gillies et al. [11-13], Govier et al. [14], it is found that attempts to solve slurry flow problems may be divided into two main categories. In the first approach, one begins from the experimental facts and generalizes known correlations for some parameters by dimensional analysis, without providing an insight into the flow mechanism (e.g., [19,24,31]; and many others). In the second approach, one starts from the basic equations of motion and numerically solves these for some situations with physical or mathematical assumptions for different terms.

Numbers of such phenomenological modeling have been proposed by Wilson [27,28] Wilson and Pugh [29], Roco and Shook [20,21], Doron et al. [7] and many others.

Both of the above methods have their own limitations generated out of inherent complexity and poor understanding of two-phase flow systems. A successful predictive model with sound understanding of the fundamentals of particle laden turbulent flow, including all significant interactions and the ability to integrate these quantitatively, has not been developed until today as seen from various literatures.

Pressure drop is the most significant parameter for solid liquid transport in mineral and other solids handling industry. Power consumption and subsequently the whole economics of the hydro-transport depend on it. For this reason, pipeline design is mainly based on optimization in pressure drop and initial investment. In spite of the lack of detailed fundamental knowledge required for the formulation and modeling of multiphase turbulent flows, the need to predict slurry behavior handled in various industries has motivated

work, aimed at obtaining approximate solutions. Various investigations ([7,9,13,26,27], etc.) have tried to show and propose correlation relating pressure drop with other parameters of solid-liquid flow, namely solids density, liquid density, particle size, concentration, pipe diameter, viscosity of flowing media, velocity of suspension etc.

To facilitate the design and scale up of pipelines and slurry pumps, there is a need for a correlation that can predict slurry pressure drop over a wide range of operating conditions, physical properties and particle size distributions. Industry needs quick and easily implemented solutions. The model derived from the first principle is no doubt the best solution. But in the scenario where the basic principles for pressure drop modeling accounting all the interactions for slurry flow are absent, the numerical model may be promising to give some quick, easy solutions for slurry pressure drop prediction.

This paper presents a systematic approach using robust hybrid SVR-DE techniques to build a pressure drop correlation from available experimental data. This correlation has been derived from a broad experimental data bank collected from the open literature (800 measurements covering a wide range of pipe dimensions, operating conditions and physical properties).

2. Development of the Support Vector Regression (SVR) Based Correlation

The development of the SVR-based correlation was started with the collection of a large databank. The next step was to perform a support vector regression, and to validate it statistically.

2-1. Collection of Data

As mentioned earlier, over the years researchers have amply quantified the pressure drop of slurry flow in pipeline. In this work, about 220 experimental points have been collected from 20 sources spanning the years 1977-2000. This wide range of database includes experimental information from different physical systems to provide a unified correlation for pressure drop. Table 1 suggests the wide range of the collected databank for pressure drop.

2-2. Identification of Input Parameters

After an extensive literature survey, all physical parameters that influence pressure drop are put in a so-called 'wish-list'.

Out of the number of inputs in 'wish list', we used support vector regression to establish the best set of chosen inputs, which describes pressure drop. The following criteria guide the choice of the set of inputs:

- The inputs should be as few as possible.
- Each input should be highly cross-correlated with the output parameter.
- These inputs should be weakly cross-correlated with each other.
- The selected input set should give the best output prediction, which is checked by using statistical analysis [e.g., average absolute relative error (AARE), standard deviation].

While choosing the most expressive inputs, there is a compromise between the number of inputs and prediction. Based on different combinations of inputs, trial and error method was used to finalize the input set which gives reasonable low prediction error (AARE) when exposed to support vector regression.

Based on the above analysis, the input variables such as pipe diameter, particle diameter, solids concentration, solid and liquid density and viscosity of flowing medium have been finalized to predict pressure drop in slurry pipeline. Table 2 shows some typical data used for support vector regression.

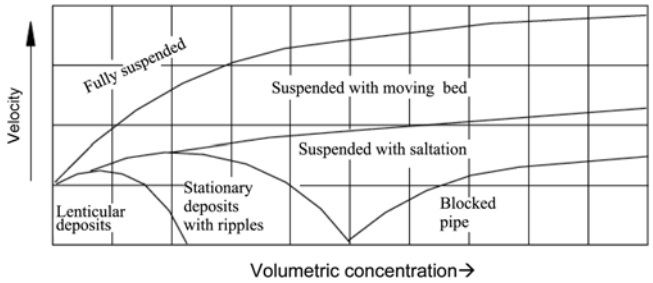

Fig. 2. Flow regimes in terms of velocity versus volumetric concentration [19].

Table 2. Different loss function

Case	Name of loss function
Case 1	ϵ-insensitive loss function
Case 2	Quadratic loss function

RESULTS AND DISCUSSION

As the magnitude of inputs and outputs greatly differ from each other, they are normalized in −1 to +1 scale. 75% of total dataset was chosen randomly for training and the rest, 25%, was selected for validation and testing.

Seven parameters were identified as input (Table 4) for SVR and

Table 3. System and parameter studied [7,9,10,12,18,20]

Slurry system: Coal water, Copper ore water, Sand water, Gypsum water, Glass water and Gravel water	
Pipe diameter (m)	0.019-0.495
Particle diameter (m)*10^{-6}	38.3-13000
Liquid density (kg/m^3)	1000-1250
Solids density (kg/m^3)	1370-2844
Liquid viscosity (Pa-m)*10^{-3}	0.12-4
Velocity (m/s)	0.86-4.81
Solids concentration (volume fraction)	0.014-0.333

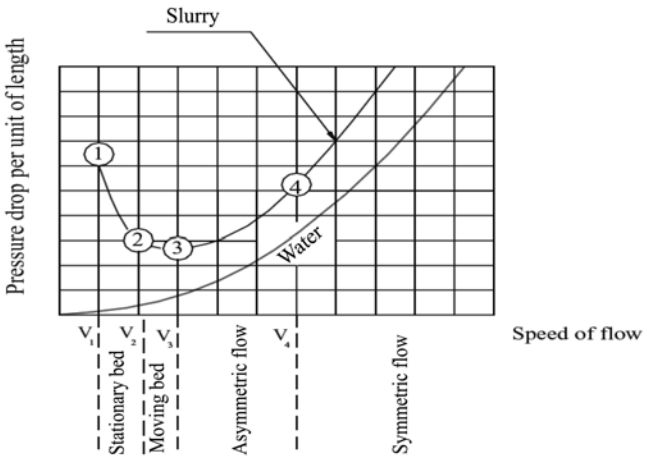

Fig. 3. Plot of transitional mixture velocity with pressure drop.

Table 4. Typical input and output data for SVR training

Input							Output
Pipe dia (cm)	Particle dia (micron)	Liquid density (g/cc)	Solid density (g/cc)	Liquid viscosity (cp)	Slurry velocity (m/s)	Solid conc. (vol. frac.)	Dpdz (Pa/m)
2.54	40.1	1.00	2.84	0.85	1.06	0.07	560.00
2.54	40.1	1.00	2.84	0.85	1.13	0.07	600.00
2.54	40.1	1.00	2.84	0.85	1.15	0.07	600.00
1.90	40.1	1.00	2.84	0.85	1.10	0.07	1250.00
1.90	40.1	1.00	2.84	0.85	1.50	0.07	1500.00
1.90	40.1	1.00	2.84	0.85	1.70	0.07	1700.00
5.26	38.3	1.00	2.33	1.00	1.11	0.11	294.10
5.26	38.3	1.00	2.33	1.00	3.01	0.11	1651.30
5.26	38.3	1.00	2.33	1.00	4.81	0.11	3822.90
5.26	38.3	1.00	2.33	1.00	1.33	0.31	542.90
5.26	38.3	1.00	2.33	1.00	3.12	0.31	2352.60
5.26	38.3	1.00	2.33	1.00	4.70	0.31	4727.70
20.85	190.0	1.00	1.37	1.14	2.59	0.33	266.50
20.85	190.0	1.00	1.37	1.14	2.34	0.33	226.30
20.85	190.0	1.00	1.37	1.14	2.01	0.33	177.30
20.85	190.0	1.00	1.37	1.14	1.78	0.33	147.00
20.85	190.0	1.00	1.37	1.14	1.59	0.32	123.40
20.85	190.0	1.00	1.37	1.14	1.37	0.33	99.90
5.15	165.0	1.00	2.65	1.00	1.66	0.07	666.20
5.15	165.0	1.00	2.65	1.00	3.78	0.09	2449.20
5.15	165.0	1.00	2.65	1.00	1.66	0.17	901.30
5.15	165.0	1.00	2.65	1.00	4.17	0.19	3428.90
5.15	165.0	1.00	2.65	1.00	1.66	0.27	1136.40
5.15	165.0	1.00	2.65	1.00	4.33	0.29	4408.10
26.30	165.0	1.00	2.65	1.00	2.90	0.09	261.60
26.30	165.0	1.00	2.65	1.00	3.50	0.09	334.10
26.30	165.0	1.00	2.65	1.00	2.90	0.18	305.70
26.30	165.0	1.00	2.65	1.00	3.50	0.17	382.10
26.30	165.0	1.00	2.65	1.00	2.90	0.26	355.60
26.30	165.0	1.00	2.65	1.00	3.50	0.26	453.60
26.30	165.0	1.00	2.65	1.00	2.90	0.33	414.40
26.30	165.0	1.00	2.65	1.00	3.50	0.32	526.10
49.50	165.0	1.00	2.65	1.00	3.16	0.09	143.00
49.50	165.0	1.00	2.65	1.00	3.76	0.09	186.10
49.50	165.0	1.00	2.65	1.00	3.07	0.17	157.70
49.50	165.0	1.00	2.65	1.00	3.76	0.17	210.60
49.50	165.0	1.00	2.65	1.00	3.16	0.26	193.00
49.50	165.0	1.00	2.65	1.00	3.76	0.26	254.70
15.85	190.0	1.00	2.65	1.30	2.50	0.14	475.20
15.85	190.0	1.00	2.65	1.30	2.50	0.29	630.90
15.85	190.0	1.00	2.65	0.12	3.00	0.13	648.90
15.85	190.0	1.00	2.65	0.12	2.90	0.28	866.70
5.07	520.0	1.00	2.65	1.00	1.90	0.09	1175.60
5.07	520.0	1.00	2.65	1.00	2.00	0.21	1763.40
4.00	580.0	1.25	2.27	4.00	2.88	0.16	3926.00
4.00	580.0	1.25	2.27	4.00	2.70	0.14	3580.00
4.00	580.0	1.25	2.27	4.00	2.01	0.10	2217.00
4.00	580.0	1.25	2.27	4.00	1.05	0.06	845.00
26.30	13000.0	1.00	2.65	1.00	3.20	0.04	842.50
26.30	13000.0	1.00	2.65	1.00	4.00	0.04	989.50

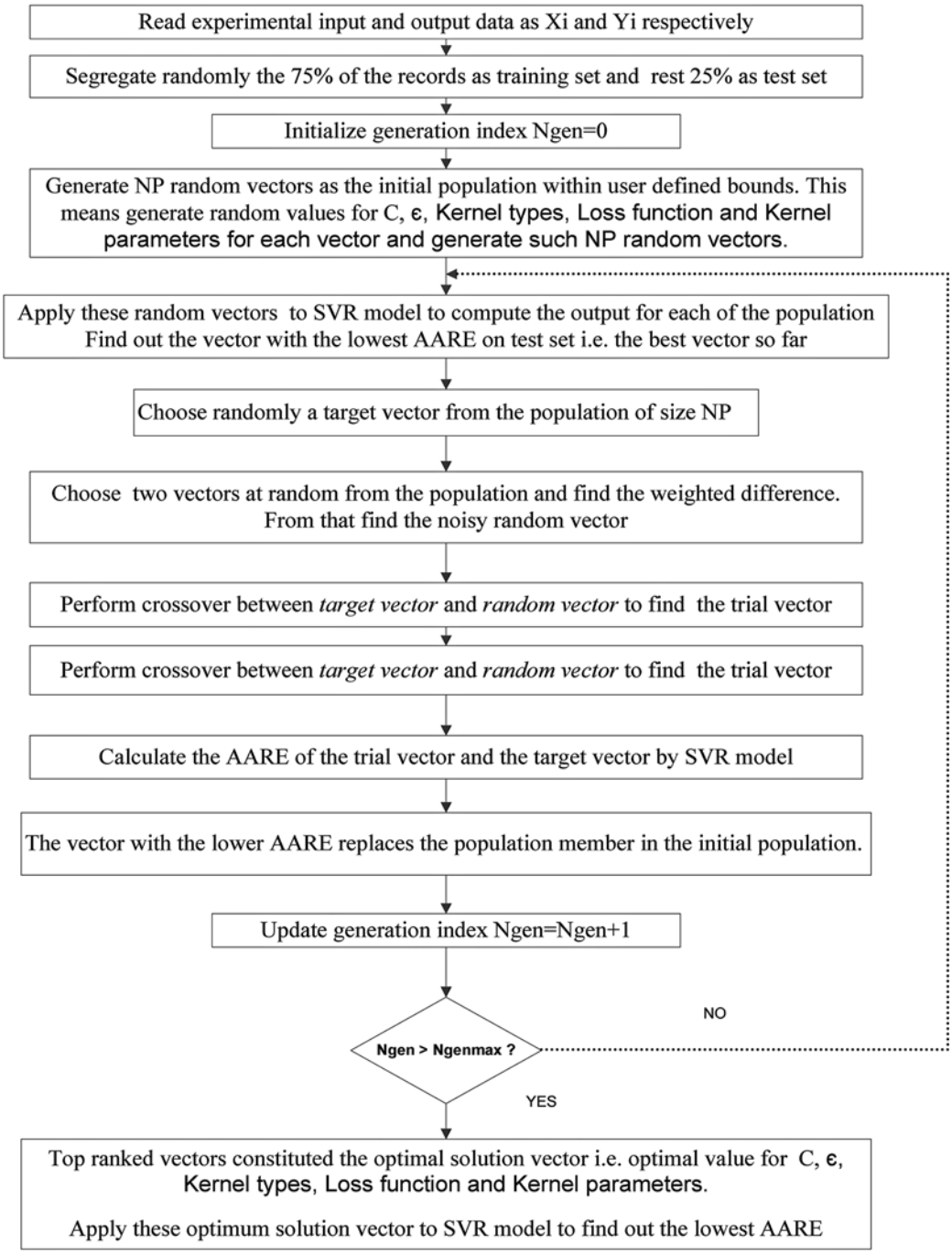

Fig. 4. Schematic for SVR algorithm implementation.

Table 5. Prediction error by SVR based model

Prediction performance by SVR		
	Training	Testing
AARE	0.125	0.127
Sigma	0.163	0.169
R	0.978	0.977

the pressure drop was put as target. These data then were exposed to the hybrid SVR-DE model described above. After optimization of five SVR parameters described above, the model output was summarized in Table 5. The low AARE (12.7%) may be considered as a excellent prediction performance considering the poor understanding of slurry flow phenomena and large databank for training comprising various systems. The optimum value of SVR meta parameters are summarized in Table 6. From Table 6 it is clear that there exist almost five different feasible solutions which lead to the same prediction error.

In a separate study, we exposed the same dataset to the SVR algorithm only (without the DE algorithm) and tried to optimize the different parameters based on exhaustive search. We found that it

Table 6. Optimum parameters obtained by hybrid SVR-DE algorithm

Sr No	C	e	Kernel type	Type of loss function	Kernel parameter	AARE
1	5043.82	0.50	erbf	e-insensitive	2	0.127
2	6698.16	0.36	erbf	e-insensitive	3	0.127
3	7954.35	0.48	erbf	e-insensitive	4	0.127
4	8380.98	0.54	erbf	e-insensitive	5	0.127

Table 7. Comparison of performance of SVR-DE hybrid model Vs SVR model

Performance criteria	Prediction performance by hybrid SVR-DE model	Prediction performance by SVR model without DE tuning after exhaustive search
	Testing	Testing
AARE	0.127	0.132
Sigma	0.169	0.170
R	0.977	0.975
Execution time (hr)	1	4

Table 8. Performance of different correlations to predict pressure drop

Sl no	Author	% AARE
1	Wilson (1942)	49.51
2	Durand and Condolios (1952)	36.53
3	Newitt et al. (1955)	93.43
4	Zandi and Govatos (1967)	50.02
5	Shook (1969)	34.50
6	Turian and Yuan (1977)	39.97
7	Wasp et al. (1977)	26.68
8	Gillies et al. (1999)	22.31
9	Kaushal (2002)	22.01
10	Present work	12.70

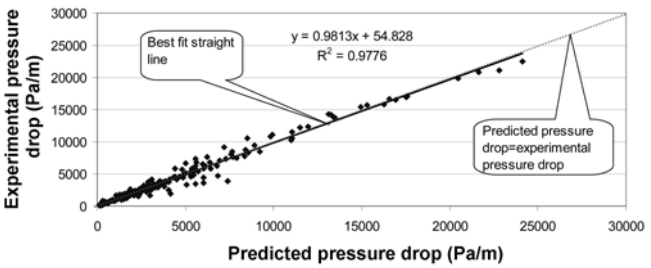

Fig. 5. Experimental Vs predicted pressure drop for erbf kernel.

was not possible to reach the best solutions starting from arbitrary initial conditions. In particular, the optimum choice of C and € is very difficult to arrive after starting with some discrete value. Many times the solutions got stuck in sub optimal local minima. These experiments justified the use of a hybrid technique for SVR parameter tuning. The best prediction after exhaustive search along with SVR parameters is summarized in Table 7. From the table it is clear that even after 720,000 runs, the SVR algorithm is unable to locate the global minima and the time of execution is 4 hrs on a Pentium 4 processor. On the other hand, the hybrid SVR-DE technique is able to locate the global minima with 2000 runs within 1 hr. The prediction accuracy is also much better. Moreover, it relieves the non-expert users to choose the different parameters and find optimum SVR meta parameters with a good accuracy.

All the 800 experimental data collected from open literature was also exposed to different formulas and correlations for pressure drop available in the open literature and AARE were calculated for each of them (Table 8). From Table 8, it is evident that the prediction error of pressure drop has reduced considerably in the present work.

CONCLUSION

Support vector machine regression methodology with a robust parameter tuning procedure has been described in this work. The method employs a hybrid SVR-DE approach for minimizing the generalization error. Superior prediction performances were obtained for the case study of pressure drop, and a comparison with selected correlations in the literature showed that the developed SVR correlation noticeably improved prediction of pressure drop over a wide range of operating conditions, physical properties, and pipe diameters. The proposed hybrid technique (SVR-DE) also relieves the non-expert users to choose the meta parameters of the SVR algorithm for their case study and find the optimum value of these meta parameters on their own. The results indicate that SVR-based technique with the DE based parameters tuning approach described in this work can yield excellent generalization and can be advantageously employed for a large class of regression problems encountered in process engineering.

NOMENCLATURE

α, α^* : vectors of Lagrange's multiplier
ε : precision parameter
ε_{loss} : loss function parameter
ε_{tol} : tolerance for termination criterion
σ : width of kernel of radial basis function
l : regularization constant
ξ, ξ^* : slack variables

REFERENCES

1. M. Agarwal, A. M. Jade, V. K. Jayaraman and B. D. Kulkarni, *Chem. Eng. Prog*., 57 (2003).
2. B. V. Babu and K. K. N. Sastry, *Comp. Chem. Eng*., **23**, 327 (1999).
3. C. Burges, *Data Mining and Knowledge Discovery*, **2**(2), 1 (1998).
4. H. M. Cartwright and R. A. Long, *Ind. Eng. Chem. Res*., **32**, 2706 (1993).
5. V. Cherkassky and F. Mulier, *Learning from data: Concepts theory and methods*, John Wiley & Sons (1998).
6. V. Cherkassky and Y. Ma, *Practical selection of SVM parameters and noise estimation for SVM regression*, Neurocomputing (special issue on SVM) (2002).
7. P. Doron, D. Granica and D. Barnea, *Int. J. of Multiphase Flow*, **13**, 535 (1987).
8. T. F. Edgar and D. M. Himmelblau, *Optimization of chemical processes*, McGraw- Hill, Singapore (1989).
9. A. Garrard and E. S. Fraga, *Comput. Chem. Eng*., **22**, 1837 (1988).
10. K. C. Ghanta, *Studies on rhelogical and transport characteristic of solid liquid suspension in pipeline*, PhD Thesis, IIT Kharagpur (1996).
11. R. G. Gillies, K. B. Hill, M. J. Mckibben and C. A. Shook, *Powder Technology*, **104**, 269 (1999).
12. R. G. Gillies and C. A. Shook *The Can. J. Chem. Eng.*, **78**, 709 (2000).
13. R. G. Gillies, C. A. Shook and K. C. Wilson, *The Canadian Journal of Chemical Engineering*, **69**, 173 (1991).
14. G. W. Govier and K. Aziz, *The flow of complex mixtures in pipes*, Krieger Publication, Malabar, FL (1982).
15. T. Hastie, R. Tibshirani and J. Friedman, *The elements of statistical learning data mining inference and prediction*, Springer (2001).
16. L. B. Jack and A. K. Nandi, *Mech. Sys. Sig. Proc*., **16**, 372 (2002).
17. D. R. Kaushal and Tomita Yuji, *Int. J. Multiphase Flow*, **28**, 1697 (2002).
18. D. Mattera and S. Haykin, *Support vector machines for dynamic reconstruction of a chaotic system*, Advances in Kernel Methods, Support Vector Machine, MIT Press (1999).
19. D. M. Newitt, J. F. Richardson, M. Abbott and R. B. Turtle, *Trans. Inst. Chem. Eng.*, **33**, 93 (1955).
20. K. Price and R. Storn, *Differential evolution*, Dr. Dobb's J., 18-24 (1997).
21. M. C. Roco and C. A. Shook, *Powder Technology*, **39**, 159 (1984).
22. M. C. Roco and C. A. Shook, *J. Fluids Eng*., **107**, 224 (1985).
23. K. K. N. Sastry, L. Behra and I. J. Nagrath, *Differential evolution based fuzzy logic controller for nonlinear process control*, Fundamenta Informaticae, Special Issue on Soft Comput. (1998).
24. B. Scholkopf, J. Burges and A. Smola, *Advances in kernel methods: Support vector machine*, MIT Press (1999).
25. B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola and R. C. Williamson, *Neural Comput*., **13**, 1443 (2001).
26. A. Smola, N. Murata, B. Schölkopf and K. Muller, *Asymptotically optimal choice of epsilon-loss for support vector machines*, Proc. ICANN (1998).
27. R. M. Turian and T. F. Yuan, *AIChE J*., **23**, 232 (1977).
28. V. Vapnik, S. Golowich and A. Smola, *Adv. in Neural Inform. Proces. Syst*., **9**, 281 (1996).
29. V. Vapnik, *The nature of statistical learning theory*, Springer Verlag, New York (1995).
30. V. Vapnik, *Statistical learning theory*, John Wiley, New York (1998).
31. E. J. Wasp, T. C. Aude, J. P. Kenny, R. H. Seiter and R. B. Jacques, *Deposition velocities transition velocities and spatial distribution of solids in slurry pipelines*, Proc. Hydro transport 1, BHRA Fluid Engineering, Coventry, UK, paper H42 53-76 (1970).
32. K. C. Wilson, *A unified physically-based analysis of solid-liquid pipeline flow*, Proceedings of the 4th International Conference on Hydraulic Transport of Solids, BHRA Fluid Engineering, Cranfield UK Paper A2, 1-16 (1976).
33. K. C. Wilson and F. J. Pugh, *Can. J. Chem. Eng*, **66**, 721 (1988).
34. I. Zandi and G. Govatos, *Proc. ACSE, J. Hydraul. Div.*, **93**(HY3), 145 (1967).