Internet Electronic Journal of Molecular Design **2004**, *3*, 118–133 ISSN 1538–6414
BioChem Press http://www.biochempress.com

Inter*net* Electronic Journal of Molecular Design

March 2004, Volume 3, Number 3, Pages 118–133

Editor: Ovidiu Ivanciuc

Special issue dedicated to Professor Nenad Trinajstić on the occasion of the 65th birthday
Part 9

Guest Editor: Douglas J. Klein

Structure–Activity Relationships using Locally Linear Embedding Assisted by Support Vector and Lazy Learning Regressors

Rakesh Kumar,[1] Abhijit Kulkarni,[1] Valadi K. Jayaraman,[1] and Bhaskar D. Kulkarni[1]

[1] Chemical Engineering Division, National Chemical Laboratory, Pune 411008, India

Received: September 1, 2003; Revised: February 10, 2004; Accepted: March 8, 2004; Published: March 31, 2004

Citation of the article:
R. Kumar, A. Kulkarni, V. K. Jayaraman, and B. D. Kulkarni, Structure–Activity Relationships using Locally Linear Embedding Assisted by Support Vector and Lazy Learning Regressors, *Internet Electron. J. Mol. Des.* **2004**, *3*, 118–133, http://www.biochempress.com.

Inter*net* Electronic Journal of Molecular Design

BioChem Press
http://www.biochempress.com

Structure–Activity Relationships using Locally Linear Embedding Assisted by Support Vector and Lazy Learning Regressors[#]

Rakesh Kumar,[1] Abhijit Kulkarni,[1] Valadi K. Jayaraman,[1] and Bhaskar D. Kulkarni[1,*]

[1] Chemical Engineering Division, National Chemical Laboratory, Pune 411008, India

Received: September 1, 2003; Revised: February 10, 2004; Accepted: March 8, 2004; Published: March 31, 2004

Internet Electron. J. Mol. Des. **2004**, *3* (3), 118–133

Abstract

Motivation. Structure–activity relationships are characterized by large dimensions and conventional procedures become protracted while modeling these relationships. To enhance the modeling abilities in terms of reduced computational costs motivates the use of recently developed tools in machine learning.

Method. Newly developed locally linear embedding is used in reducing the nonlinear dimensions in QSPR and QSAR. The reduced set is subsequently modeled with robust regressors, namely lazy learning and support vector regression.

Results. Both the datasets show improved results with the reduced dimensions as compared to their original dimension counterparts.

Conclusions. Locally linear embedding for nonlinear dimensionality reduction coupled with robust regressors such as lazy learning and support vector regression seems to be a promising option in analyzing the nonlinear datasets.

Keywords. Locally linear embedding; lazy learning; support vector regression; QSPR; quantitative structure–property relationships; QSAR; quantitative structure–activity relationships.

Abbreviations and notations

LLE, locally linear embedding	RBF, radial basis function
MSE, mean squared error	QSAR, quantitative structure–activity relationships
PCA, principal component analysis	QSPR, quantitative structure–property relationships
PRESS, prediction sum of squares	SVM, support vector machine
QP, quadratic programming	SVR, support vector regression

1 INTRODUCTION

Development of structure–activity relationships (QSPR and QSAR) is one of the most common problems involved in development of new materials with desirable properties. The problem is encountered in synthesis of advanced materials such as a catalyst, polymer composites,

[#] Dedicated to Professor Nenad Trinajstić on the occasion of the 65th birthday.

* Correspondence author; phone: +91–20–5893095; fax: +91–20–5893041; E–mail: bdk@ems.ncl.res.in.

pharmaceutical data analysis for drug design etc. The conventional approach uses variations of the classical multivariate regression techniques and principal component analysis. More recently neural network based paradigms have been used to capture the highly nonlinear and complex structure–activity relationships [1]. Owing to certain disadvantages like long training times, overfitting the training data etc., researchers are continuously looking for more accurate and informative techniques [1–3]. The present work explores combinations of some of the newly developed computational tools in machine learning to build robust QSPR and QSAR models. More specifically, the recently developed Locally linear embedding technique has been employed for reducing the dimensionality of data which is subsequently modeled with (*a*) adaptive memory based local learning (*i.e*. lazy learning) and (*b*) support vector regression techniques.

The paper is organized as follows. First we explain the chemical data considered followed by basic locally linear embedding, lazy learning and support vector regression algorithms in Section 2. Section 3 on results and discussion confers the salient features of the algorithm. Finally, section 4 on conclusions summarizes the results.

2 MATERIALS AND METHODS

2.1 Chemical Data

Datasets considered in the analysis are taken from literature and include the quantitative structure property relationships for predicting boiling points of alkanes and quantitative structure activity relationships for predicting biological activity. We give a brief account of these datasets.

2.1.1 Quantitative structure property relationship data

QSPRs correlate and estimate the physical properties of the organic compounds. Specifically, they correlate physicochemical properties with molecular structural characteristics (geometric and electronic) expressed in terms of appropriate molecular descriptors. Various descriptors such as electronic (dipole moments etc.), lipophilic (partition coefficients), topological (molecular connectivity indices and other geometric parameters) as well as molecular parameters (molar volume, parachor, etc.) can be used for correlating structural parameters with physicochemical properties. The data used in the present work is taken from Trinajstić *et al*. [4] wherein different descriptors based on distance indices and two connectivity indices for the first 150 alkanes are reported [5–6].

2.1.2 Quantitative structure activity relationship data

QSAR analysis is based on the assumption that there exists relationship between the biological activity within a group of molecular compounds with the variation of their respective structural and chemical features. So based on these physicochemical descriptors, analyst tries to predict the

molecule's activity. The benchmark data set under consideration is taken from Selwood *et al*. [7] where they developed a series of analogues of the mammalian electron transport inhibitor antimycin A_1 as potential anitfilarials in order to design new antifilarial drugs.

The other set under consideration is steroids data set. It mainly deals with a problem to model the corticosteroid binding globulin (CBG) receptor activity with autocorrelation of molecular surface properties. Wagener *et al*. [8] modified this particular dataset from its earlier version to form in all 31 steroids with 1248 descriptors to model CBG receptor activity.

With this, we now briefly describe the algorithms employed in this work to develop data driven models for predicting QSARs and QSPRs. We first provide a gist of the basic locally linear embedding algorithm for nonlinear dimensionality reduction. Finally we proceed to give an outline of the two frequently used robust regression algorithms, namely lazy learning and support vector regression.

2.2 Locally Linear Embedding (LLE) Algorithm

The LLE algorithm is based on simple geometrical considerations [9]. Let the data set $\{\vec{X}\}_{i=1,2.....N} \in \Re^D$ be sampled from some smooth underlying manifold. Provided the manifold is well sampled (*i.e*., there is enough data), we expect each data point and its neighbors lie on or close to a locally linear patch of the manifold. Thus we can approximate the non–linear manifold in the vicinity of X_i by the linear hyperplane passing through its nearest neighbors.

Based on this simple idea, the LLE algorithm consist of a three step process:

(1) Identify *K* nearest neighbors for every data point, as measured by Euclidean distance (Other notions of "closeness" are also possible, such as all points within a certain radius, or by using more sophisticated rules based on local metrics.)

(2) Compute the weights W_{ij} that best linearly reconstruct each X_i from its *K* neighbors. Here we are required to minimize the reconstruction error as measured by the cost function

$$\varepsilon(W) = \sum_i \left| \vec{X}_i - \sum_j W_{ij} \vec{X}_j \right|^2 \tag{1}$$

subject to two constraints:

(*a*) Each data point $\vec{X}_i$ is reconstructed only from its neighbors, enforcing $W_{ij} = 0$ if $\vec{X}_i$ does not belong to this set and

(*b*) $\sum_j W_{ij} = 1$ for every *i*.

The weights W_{ij} signify the contribution of the j^{th} data point to the i^{th} reconstruction. The optimal weights W_{ij} subject to these constraints are found by solving a least squares problem, as discussed in

Appendix 1.

The constrained weights that minimize these reconstruction errors characterize intrinsic geometric properties of each neighborhood, as opposed to properties that depend on a particular frame of reference. This is due to the fact that for any particular data point, the weights are invariant to rotations, rescalings, and translations of that data point and its neighbors. The invariance to rotations and rescalings results from the form of Eq. (1); the invariance to translations is imposed by the sum–to–one constraint (*b*).

Since the data lie on or near a smooth nonlinear manifold of dimensionality $d << D$, there exists a linear mapping (comprising a translation, rotation, and rescaling) that maps the high dimensional coordinates of each neighborhood to global internal coordinates on the manifold. Thus reconstruction weights W_{ij}, invariant to such transformations, should characterize the local geometry to same extent, both in the original data space and local patches on the manifold. In particular, the same weights W_{ij} that reconstruct the i^{th} data point in D dimensions should also reconstruct its embedded manifold coordinates in d dimensions. This forms the basis of third step of algorithm.

(3) Find the *d*–dimensional vectors $\vec{Y}_i$ that best match those reconstruction weights *W*. Here we are required to minimize the reconstruction errors as measured by embedding cost function:

$$\Phi(Y) = \sum_i \left| \vec{Y}_i - \sum_j W_{ij} \vec{Y}_j \right|^2 \qquad (2)$$

To ensure the uniqueness of the solution the following two constraints are imposed: translation invariance by requiring the coordinates to be centered on the origin *i.e.* $\sum_i \vec{Y}_i = 0$ and we constrain the embedding vectors to have unit covariance,

$$\frac{1}{N} \sum_i \vec{Y}_i . \vec{Y}_i^{\,T} = I$$

where *I* is the $d \times d$ identity matrix.

These constraints do not affect the generality of the solutions as $\Phi(Y)$ is invariant to translation, rotations and homogeneous rescalings. The additional constraint that the covariance is equal to the identity matrix expresses an assumption that reconstruction errors for different coordinates in the embedding space should be measured on the same scale.

The optimal embedding $\vec{Y}_{i=1,2.....N} \in R^d$ is given by eigenvectors associated with the smallest d +1 eigenvalues of the matrix *M* [10]:

$$M = (I - W)^T (I - W) \qquad (3)$$

The bottom eigenvector of this matrix is discarded, as it is a vector composed of all ones, with zero as eigenvalue. Discarding this eigenvector enforces the constraint that the embeddings have zero mean, as the components of other eigenvectors must sum to zero, by virtue of orthogonality. The remaining *d* eigenvectors of size *N* give the final embedding *Y*.

Although the reconstruction weights for each data point are computed from its local neighborhood independently, the embedding coordinates are computed by an $N \times N$ eigen solver, a global operation that couples all data points in connected components of the graph defined by the weight matrix. The different dimensions in the embedding space can be computed successively; this is done simply by computing the bottom eigenvectors from Eq. (2) one at a time.

In situations where data is not available in vector form $\vec{X}_i$, but only as the measurements of dissimilarity or pair wise distance between different data points, a simple variation of LLE can be applied. The nearest neighbors are identified by the smallest non–zero elements of each row in the distance matrix. As described in Appendix 1, the reconstruction weights calculation for each data point requires computing the local covariance matrix C_{jk} between its nearest neighbors. This is done by exploiting the usual relation between pair wise distances and dot products that form the basis of metric MDS [11]. Therefore, for a particular data point

$$C_{jk} = \frac{1}{2}(D_j + D_k - D_{jk} - D_0) \tag{4}$$

where D_{jk} is the squared distance between the j^{th} and i^{th} neighbors,

$$D_\ell = \sum_z D_{\ell z} \text{ and } D_0 = \sum_{jk} D_{jk}$$

The rest of the algorithm proceeds as usual. The procedure as described above leads to nonlinear dimensionality reduction of data.

2.2.1 Parameters

In order to get a good LLE mapping, two parameters viz. the dimensionality, *d*, and the number of neighbors, *K*, need attention. If *d* is set too high, the mapping will enhance noise (due to the constraint $1/n\mathrm{YY}^{\mathrm{T}} = \mathbf{I}$); if it is set too low, distinct parts of the data set might be mapped on top of each other. If *K* is set too small, the mapping will not reflect any global properties; if it is too high, the mapping will lose its nonlinear character and behave like traditional PCA, as the entire data set is seen as local neighborhood. In the present work dimensionality *d* is chosen according to the criteria: $d + 1$ should be less than or equal to the number of eigen values of *M* that are close to zero [10–11].

The nearest neighbor parameter *K* is a measure of the "quality" of input–output mapping (*i.e.*, how well the high–dimensional structure is represented in the embedded space) and is selected

based on the residual variance [12]. It is defined as $1-\rho^2_{D_xD_y}$ where ρ is the standard linear correlation coefficient, computed over all entries of D_x and D_y; D_x and D_y are the matrices of Euclidean distances (between pairs of points) in X and Y (Y was computed in Step 3 above), respectively. The lower the residual variance is, the better the high–dimensional data are represented in the embedded space. Hence, the optimal value for K, K_{opt} can be determined as:

$$K_{opt} = \arg\min_K (1-\rho^2_{D_xD_y}) \tag{5}$$

Although a few techniques are given in literature [13] like linear interpolations and training a neural network or RBF network for mapping a new (previously unseen) sample, we have preferred a simple strategy of concatenating the new sample with given samples and repeating the whole LLE procedure. This preference is based on our observation that the LLE algorithm takes only few seconds of time to run, retaining non–linear mapping even for query point. Whereas the approaches like Neural or RBF networks are hard to train and linear interpolations will lose the non– linearity of data.

2.3 Lazy Learning

Lazy learning is a memory based local learning method using local selection of parameters [14]. It is a query based technique and defers all the computations till a query is received for prediction. It answers the query point by interpolating locally the relevant examples according to a distance measure. The local modeling procedure used in prediction of the query point consists of parametric and structural identification. Given a model structure, parametric identification optimizes the parameters of the local approximator. The structural identification, on the other hand, selects a family of local appproximators, a metric to evaluate relevant examples and *bandwidth* that indicates the size of the region in which the data are correctly modeled by the members of the local approximators chosen earlier [15]. Being a memory–based technique, separate training is not required to answer the query points, which greatly improves the speed of implementation. Secondly, it predicts by locally interpolating the relevant points based on a distance measure. This is particularly useful when limited amount of input/output data is available and an accurate prediction is required. Lastly, it is less susceptible to the noise contamination. All these features greatly improve the overall performance of the learning from input/output data as compared to other parametric as well as nonparametric methods. A brief introduction on the working of algorithm is as follows.

Consider two variables $\mathbf{x} \in \Re^m$ and $y \in \Re$ for unknown input–output mapping $f: \Re^m \to \Re$ known through a set of n examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$. Let this mapping be represented as:

$$y_i = f(\mathbf{x}_i) + \varepsilon_i \tag{6}$$

where $\forall i$, ε_i is a random variable such that $E[\varepsilon_i] = 0$ and $E[\varepsilon_i \varepsilon_j] = 0$, $\forall j \neq i$, and such that

$E\left[\varepsilon_i^r\right]=\mu_r\left(\mathbf{x}_i\right), \forall r \geq 2$, where $\mu_r(\cdot)$ is the unknown r^{th} moment of distribution of ε_i and is defined as a function of $\mathbf{x}_i$.

Now given a query point $\mathbf{x}_q$, the parameter β of a local linear approximation of $f(\cdot)$ in a neighborhood of $\mathbf{x}_q$ is obtained by solving the local polynomial regression:

$$\sum_{i=1}^{n}\left\{(y_i-\mathbf{x}_i'\beta)^2\, K\left(\frac{D(\mathbf{x}_i,\mathbf{x}_q)}{h}\right)\right\} \tag{7}$$

where, given a metric on the space $\Re^m$, d ($\mathbf{x}_i$, $\mathbf{x}_q$) is the distance from the query point to the i^{th} example, K (.) is weight function, and h is the bandwidth. To consider a constant term in the regression, a constant value of 1 is appended to each input vector $\mathbf{x}_i$.

The solution of the above weighted least squares problem to find β is

$$\hat{\beta}=\left(X'W'W\,X\right)^{-1}X'W'W\mathbf{y} \tag{8}$$

where X is a matrix whose i^{th} row is $\mathbf{x}_i'$, $\mathbf{y}$ is a vector whose i^{th} element is y_i, W is a diagonal matrix whose i^{th} diagonal element is

$$w_{ii}=\sqrt{K\left(d(\mathbf{x}_i,\mathbf{x}_q)/h\right)} \tag{9}$$

Replacing, $Z = X\,W$ and $v = W\,\mathbf{y}$ in Eq. (8)

$$\hat{\beta}=\left(Z'Z\right)^{-1}Z'v=PZ'v \tag{10}$$

It is assumed that matrix P is nonsingular so that its inverse is defined.

Once the local linear polynomial approximation is obtained, a prediction of y_q = f ($\mathbf{x}_q$) is given by

$$\hat{y}_q=\mathbf{x}_q'\,\hat{\beta} \tag{11}$$

The cross–validation procedure gives the assessment of the mean–squared–error as

$$mse=E\left[(y_q-\hat{y}_q)^2\right] \tag{12}$$

Cross–validation requires a large computational effort to be performed due to the series of training steps. Instead, the PRESS statistic, which returns the leave–one–out cross validation error at the reduced computational effort, is used extensively in case of linear models. The PRESS statistic [16] is given by:

$$e_j^{CV}=y_j-\mathbf{x}_j'\,\hat{\beta}_{-j} \tag{13}$$

where e_j^{CV} is the leave–one–out–error and $\hat{\beta}_{-j}$ is the estimated regression parameter with the j^{th} sample removed from the available set of examples. Thus, PRESS statistics returns LOO cross validation error at reduced computational cost and ensures that best model is selected for prediction.

The above equation can be simplified as:

$$\frac{y_j - \mathbf{x}'_j PZ'v}{1-\mathbf{z}'_j P\mathbf{z}_j} = \frac{y_j - \mathbf{x}'_j \hat{\beta}}{1-h_{jj}} \tag{14}$$

where

$$\mathbf{z}_j = w_{jj}\,\mathbf{x}_j \tag{15}$$

which is nothing but the j^{th} row of Z.

The term h_{jj} is the j^{th} diagonal matrix of the *Hat matrix*, H, given as

$$H = Z\,P\,Z' \tag{16}$$

From Eq. (14), it is evident that it is possible to calculate leave–one–out error without having to explicitly identifying $\hat{\beta}_{-j}$. This simplification greatly reduces the computations making the lazy learning regression algorithm very robust and competitive.

The extension of this general procedure to estimate the parameters, to obtain the leave–one–out errors and to generate local models, recursively is also proposed. The recursive algorithm helps in adaptively selecting the best number of nearest neighbors by calculating the corresponding leave–one–out errors without any computational load [17–19]. Once the local models are generated recursively, the best model is selected based on either *winner–takes–all* (competitive) approach or using local combinations (cooperative) of models.

Winner–takes–all, which is adopted in the present work, selects the best approximator based on some given criterion, *mean–squared–error* (*mse*) being the most convenient and classical one. So, prediction obtained for each value of k is compared based on *mean squared error* and the final prediction is done as:

$$\hat{y}_q = \mathbf{x}'_q\,\hat{\beta}\left(\hat{k}\right) \tag{17}$$

with,

$$\hat{k} = \arg\min_{k\in S} mse^{cv}(k) \tag{18}$$

where S is a range from which the optimal number of neighbors are selected.

The form of local constant models used in the present work is as follows [15]:

$$\hat{y}_{0,q}(k) = \frac{k-1}{k}\hat{y}_{0,q}(k-1) + \frac{1}{k}y(k) \tag{19}$$

and the corresponding *mse* is calculated as:

$$mse_0^{cv}(k) = \frac{k(k-2)^2}{(k-1)^3} mse_0^{cv}(k-1) + \frac{1}{k-1}\left(y(k) - \hat{y}_{0,q}(k-1)\right)^2 \tag{20}$$

where recursion on *mse* is started for k = 2. Eqs. (19) and (20) computes the leave–one–out mean

squared error, without explicitly computing each single cross–validation error.

Lazy learning with local constant model as an approximator, incorporating Euclidean distance as a metric to find the number of nearest neighbors, is used in all the examples considered.

2.4 Support Vector Regression

The applications in which support vector machines (SVMs) have been used are wide including molecular modeling [20–21]. As the theory of support vector machines for classification and regression is well developed and a number of good research papers have been published, only a brief description of the support vector regression formalism is presented below.

In support vector regression the idea is to map the input data into a high dimensional feature space and subsequently carry out the linear regression in the feature space [22]. Thus, the input–output pairs of training data of size *l*

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_l, y_l) \qquad \mathbf{x} \in \Re^n, y \in \Re$$

can be expressed as

$$f(\mathbf{x}) = (\mathbf{w} \bullet \Phi(\mathbf{x})) + b \tag{21}$$

where **w** is a weight vector of dimension $[l \times 1]$, Φ is the function mapping the input data in the feature space, $\mathbf{w} \bullet \Phi(\mathbf{x})$ is the dot product and b is bias. **w** is calculated by minimizing [22]:

$$R_{reg}[f] = \sum_{i=1}^{l} C(f(\mathbf{x}_i) - y_i) + \lambda \|\mathbf{w}\|^2 \tag{22}$$

where $R_{reg}[f]$ is the empirical risk, $\|\mathbf{w}\|^2$ is the complexity term, λ is a regularization constant and C (.) is a cost function.

The above quadratic programming problem has a unique solution and the weight vector **w** can be expressed as the following expansion:

$$\mathbf{w} = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) \Phi(\mathbf{x}_i) \tag{23}$$

where α_i, α_i^* is the solution of the quadratic programming problem [23–25].

Substituting Eq. (23) in Eq. (21), we have,

$$f(\mathbf{x}) = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) \Phi(\mathbf{x}_i) \bullet \Phi(\mathbf{x}) + b \tag{24}$$

The difficulty of carrying out the computations in a high dimensional space is judiciously circumvented by defining an appropriate kernel function in the input space in place of the dot product in the feature space [23]. It has been shown that kernels satisfying certain requirements

(Mercer's theorem) can be defined connecting the dot products of the feature vectors to the vectors in the input space [22]:

$$K(\mathbf{x}_i, \mathbf{x}) = \Phi(\mathbf{x}_i) \bullet \Phi(\mathbf{x})$$

By substituting the kernel function in the quadratic programming equation the entire problem can be solved in the input space itself:

$$f(\mathbf{x}) = \sum_{i=1}^{s} (\alpha_i - \alpha_i^*) K(\mathbf{x}_i, \mathbf{x}) + b \tag{25}$$

where *s*, (s < the total number of input–output pairs) is the number of input data having nonzero values of (α_i, α_i^*).

This means that while we approximate any nonlinear function by training with *l* numbers of data, we can characterize the function with only *s* number of data vectors. These vectors are known as the support vectors and hence the name Support vector machines. Among the existing popular kernels, the Gaussian radial basis function (RBF) kernel is very useful and we have employed this kernel in our computations. This kernel can be defined as

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{\left\| \mathbf{x}_i - \mathbf{x}_j \right\|^2}{-2\sigma^2} \right)$$

where σ is the kernel width parameter. Vapnik's ε–insensitive loss function, which produces sparseness in the support vectors, is used in the present work and is given by:

$$\begin{aligned} C(f(\mathbf{x}) - y) &= |f(\mathbf{x}) - y| - \varepsilon, \quad for\, |f(\mathbf{x}) - y| \geq \varepsilon \\ &= 0 \qquad\qquad otherwise \end{aligned} \tag{26}$$

where ε is a prescribed parameter. This loss function does not penalize errors below some ε > 0, chosen a priori. To get good approximation, the value of ε is usually kept small (of the order of 10^{-3} to 10^{-5}).

Support vector regression minimizes the Vapnik's ε–insensitive loss function, which defines a hyperplane with width ε around the estimate [23]. The approximations, which fall within its boundary, do not contribute to the error and considered as well estimated. Those, which lie outside the tube, contribute to the loss.

With the definition of the loss function given by Eq. (26), the quadratic programming (QP) problem to find α_i, α_i^* has the following final form [23–25],

$$\min_{\alpha, \alpha^*} \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{l} \alpha_i^* (y_i - \varepsilon) - \alpha_i (y_i + \varepsilon) \tag{27}$$

subject to the constraints:

$$0 \le \alpha_i, \alpha_i^* \le \frac{1}{\lambda}, \qquad i = 1,.....,l$$

$$\sum_{i=1}^{l} (\alpha_i - \alpha_i^*) = 0$$

By solving above equation with standard QP solvers, values of α_i, α_i^* are found out. Then only the non–zero values of α_i, α_i^* are used in Eq. (25) to calculate the function value. The value of bias *b*, in Eq. (25) is calculated as [22–23]:

$$b = -\frac{1}{2}\sum_{i=1}^{l} (\alpha_i - \alpha_i^*) \left(K(\mathbf{x}_i, \mathbf{x}_r) + K(\mathbf{x}_i, \mathbf{x}_s) \right)$$

where $\mathbf{x}_r$ and $\mathbf{x}_s$ are the support vectors.

The next section describes the results obtained at length and discusses pros and cons of these algorithms.

3 RESULTS AND DISCUSSION

The algorithms explained above are used for developing data driven models for the three datasets under consideration. We first explain the methodology and follow it by providing the detailed discussion of results.

We followed two approaches. (*i*) we directly employed lazy learning and support vector regression (SVR) without resorting to LLE dimensionality reduction. (*ii*) We first used locally linear embedding (LLE) for nonlinear dimensionality reduction and subsequently predicted the properties using the regressors under consideration. For the sake of comparison, we also carried out simulations with principal component analysis (PCA). In case of QSPR dataset, root mean squared error (RMSE) was used as performance criterion in final prediction, whereas in case of QSAR datasets, usual QSAR statistics (R and S) were calculated. Randomly selected (approximately) 80% of the total data formed the training set and remaining 20% data were used as test set. Lazy learning being a memory–based technique does not require separate training. So these 20% data were used as query points. The results reported are based on average of 20 simulations (*i.e.* 20 times different query points are tried). In case of SVR, we used *n*–fold cross validation (10–fold in QSPR and 5–fold in QSAR) to minimize the training error. We have extensively used Gaussian Radial Basis Function (RBF) kernel in the simulations. The trained SVR is then used in final prediction of the test set. For the QSAR datasets, we used multiple linear regression (MLR) also. The final results for all the datasets and regressors are presented in Tables 1–5. The results reported for all the cases are based on test set. Our aim was to compare the nonlinear dimension reduction techniques and not the regressors. We were mainly concerned with the errors obtained with the data with all the features and the errors with the transformed features. The algorithm codes for analyzing these datasets were

developed in house. The data were mean centered before furnishing it to the regressors for analysis.

Table 1. Results for three individual descriptors in QSPR

No.	Regressor used	Prediction RMSE		
		2–TI	3–TI	Connectivity index
1	Lazy Learning (with local constant model)	1.8845	0.7054	0.6022
2	Support vector regression	2.1342	0.9327	0.7374

Table 2. Results for all the descriptors in QSPR

No	Regressor	Original dimension	Prediction RMSE	Reduced dimension: LLE	Prediction RMSE	Reduced dimension: PCA	Prediction RMSE
1	Lazy Learning	12	17.08	2 (Var. 0.99)	7.5381	4 (Var. 0.99)	11.124
2	Support Vector Regression	12	17.71	2 (Var. 0.99)	17.237	4 (Var. 0.99)	22.234

Table 3. Results for QSAR datasets using Multiple Linear Regression

Dataset	QSAR statistics					
	R			S		
	Without dimension reduction	Dimension reduction with LLE	Dimension reduction with PCA	Without dimension reduction	Dimension reduction with LLE	Dimension reduction with PCA
Selwood	0.77 (N×53)	0.76 (N×9)	0.76 (N×9)	0.61 (N×53)	0.60 (N×9)	0.60 (N×9)
Steroids	0.839 (N×1248)	0.821 (N×6)	0.8013 (N×28)	1.17 (N×1248)	1.19 (N×6)	2.25 (N×28)

Table 4. Results for QSAR datasets using Lazy learning (with local constant model)

Dataset	QSAR statistics					
	R			S		
	Without dimension reduction	Dimension reduction with LLE	Dimension reduction with PCA	Without dimension reduction	Dimension reduction with LLE	Dimension reduction with PCA
Selwood	0.78 (N×53)	0.79 (N×9)	0.74 (N×9)	0.64 (N×53)	0.5925 (N×9)	0.69 (N×9)
Steroids	0.8385 (N×1248)	0.8621 (N×6)	0.7925 (N×28)	1.1718 (N×1248)	1.216 (N×6)	2.315 (N×28)

Table 5. Results for QSAR datasets using Support Vector Regression

Dataset	QSAR statistics					
	R			S		
	Without dimension reduction	Dimension reduction with LLE	Dimension reduction with PCA	Without dimension reduction	Dimension reduction with LLE	Dimension reduction with PCA
Selwood	0.80 (N×53)	0.82 (N×9)	0.80 (N×9)	0.56 (N×53)	0.543 (N×9)	0.56 (N×9)
Steroids	0.848 (N×1248)	0.883 (N×6)	0.81 (N×28)	1.067 (N×1248)	1.19 (N×6)	1.287 (N×28)

QSPR data involve the use of 10 molecular descriptors based on distance indices and 2 connectivity indices, forming in all 12 features. These descriptors are used in predicting the boiling points of alkanes. Trinajstić *et al*. [4] developed the mathematical relationships between each of 12 features individually to predict the boiling points of alkanes and then compared the results among them. To test the robustness of the regressors, in our analysis, we first considered the three descriptors viz. (*i*) two dimensional topological index (2–TI) (*ii*) three dimensional topological index (3–TI) based on determinant of adjacency plus distance matrix and (*iii*) connectivity index,

individually. The details of calculating these indices are given in Trinajstić *et al*. [4]. Though it is known that in case of alkanes, $N^{0.5}$ (where N is number of carbon atoms) correlates better than topological index, we analyzed the available data in the form of topological and connectivity indices. Results for both the lazy learning and SVR are presented in Table 1. It can be seen from Table 1 that both the regressors performed well. In case of connectivity index, Trinajstić *et al*. [4] observed that this index correlates better than any other distance indices. In our analysis, we also found the same to be true with 0.6022 as RMSE in lazy learning and 0.7371 in SVR. However, in case of 3–TI, both the regressors predicted less RMSE as compared to 2–TI; exactly opposite to what Trinajstić *et al*. [4] observed.

Next in our analysis, we considered all the distance indices and connectivity indices together for predicting the boiling points. From Table 2, it is observed that combining the descriptors to predict boiling points does not help to improve the results further. This may be attributed to a large degree of variability in the data due to the presence of some unnecessary features making the data more noisy, nonlinear and difficult to correlate.

Further we applied LLE for nonlinear dimensionality reduction of data. The data set after application of the algorithm reduces to just 2 transformed features while retaining 99% variance. From Table 2, it can be seen that as compared to the data with original dimension, reduced set correlates better with both the regressors.

Even though there exists strong correlations among some of the descriptors considered in the problem [4], our results reveal that LLE helps in preserving the information content with a small number (in this case, 2 features) of transformed features. This result has the following implications. First, it may be possible to build data driven models by employing LLE even when there is a very strong intercorrelation between features. Secondly, it is very useful to employ LLE where it is difficult to extract useful information from the correlation matrix. We also carried out simulations with one of the extensively used dimension reduction technique viz. principal component analysis. From Table 2, it can be observed that PCA require more number of features to capture the same variance as captured by LLE. Moreover, the errors are found to be more as compared to LLE. This may be attributed to some information loss while reducing the dataset.

The same methodology is followed for QSAR data set also. The first set (Selwood *et al*.) consists of 53 initial physicochemical descriptor set of 31 chemical compounds. The method of computing these descriptors is described in Selwood *et al*. [7]. In this case too LLE performed well as compared to PCA (cf. Tables 3–5). However, among the regressors, all the three regressors performed almost same on all the sets with original features as well as transformed features. The QSAR statistics reported is based on test set. The second set (Steroids set) consists of 31 steroids with 1248 descriptors. The problem is to model the corticosteroid binding globulin (CBG) receptor activity with autocorrelation of molecular surface properties. In this case also, LLE found to be

performing well compared to PCA (cf. Tables 3–5). For the training set, for both the datasets, we got R–value consistently between 0.9 and 0.95 for all the regressors. LLE thus retains the important information in the data. In short, as can be seen from the results, all the regressors correlate well.

It can now be concluded that LLE performs equally well on the datasets comprising the features, which are not strongly correlated. This suitability of LLE to deal with correlated as well as decorrelated data equally well makes it appealing in QSAR and QSPR modeling problems.

In summary, LLE is found to preserve the important information in the data in a better way while simultaneously doing the nonlinear dimensionality reduction. This, when combined with robust regressors like lazy learning and SVR, can improve the performance in analyzing the nonlinear data like QSPR and QSAR with reduced computational costs.

4 CONCLUSIONS

The article presents a hybrid method for analyzing quantitative structure property and quantitative structure activity relationships. Large dimensionality characterizes these types of data. To this end, nonlinear dimensionality reduction by locally linear embedding coupled with robust regressors like lazy learning and support vector regression seems to be a promising option.

Acknowledgment

Financial support from Department of Science and Technology (DST), New Delhi is gratefully acknowledged.

Appendix 1

Optimal reconstruction weight calculation

Eq. (1) has a closed form solution [11], which determines the optimal weights W_{ij} that best reconstruct each data point from its neighbors. Suppose a particular data point $\vec{x}$ with neighbors $\vec{\psi}_{j=1,2,....K}$ and reconstruction weights $\omega_{j=1,2.....K}$ satisfying $\sum_j \omega_j = 1$.

Thus the cost function (giving reconstruction error) to be minimized is:

$$\varepsilon = \left| \vec{x} - \sum_j \omega_j \vec{\psi}_j \right|^2$$

as $\sum_j \omega_j = 1$

$$\varepsilon = \left| \sum_j \omega_j (\vec{x} - \vec{\psi}_j) \right|^2$$

Now introducing local covariance matrix

$$C_{jk} = (\vec{x} - \vec{\psi}_j).(\vec{x} - \vec{\psi}_k)$$

we have

$$\varepsilon = \sum_{jk} \omega_j \omega_k C_{jk}$$

This error can be minimized in closed form with constraint $\sum_j w_j = 1$. In terms of the inverse local covariance matrix, the optimal weights are given by:

$$\omega_j = \frac{\sum_k C_{jk}^{-1}}{\sum_{lm} C_{lm}^{-1}}$$

Thus solution involves the inversion of local covariance matrix (symmetric and semi positive definite). If the covariance matrix is singular or nearly singular (e.g. when there are more neighbors than input dimensions [$K>D$]; or when the data points are not in general position), it can be conditioned by adding a constant to the diagonal of C_{jk} (which is small relative to its trace).

5 REFERENCES

[1] S. P. Niculescu, Artificial neural networks and genetic algorithms in QSAR, *J. Mol. Struct.* (*Theochem*) **2003**, *622*, 71–83.
[2] R. Burbidge, M. Trotter, B. Buxton, and S. Holden, Drug design by machine learning: support vector machines for pharmaceutical data analysis, *Comput. Chem.* **2001**, *26*, 5–14.
[3] D.T. Manallack and D.J. Livingstone, Neural networks in drug discovery: have they lived up to their promise?, *Euro. J. Med. Chem.* **1999**, *34*, 95–208.
[4] Z. Mihalić, S. Nikolić, and N. Trinajstić, Comparative study of molecular descriptors derived from the distance matrix, *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 28–37.
[5] B. Lučić, D. Juretić, S. Nikolić, and N. Trinajstić, The Structure–Property Models Can Be Improved Using the Orthogonalized Descriptors, *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 532–538.
[6] M. Randić and N. Trinajstić, Comparative structure–property studies: The connectivity basis, *J. Mol. Struct.* (*Theochem*) **1993**, *284*, 209–221.
[7] D. L. Selwood, D. J. Livingstone, J. C. W. Comley, A. B. O'Dowd, A. T. Hudson, P. Jackson, K. S. Jandu, V. S. Rose, and J. N. Stables, Structure–Activity Relationships of Antifilarial Antimycin Analogues: A Multivariate Pattern Recognition Study, *J. Med. Chem.* **1990**, *33*, 136–142.
[8] M. Wagener, J. Sadowski, and J. Gasteiger, Autocorrelation of Molecular Surface Properties for Modeling Corticosteroid Binding Globulin and Cytosolic Ah Receptor activity by Neural Networks, *J. Am. Chem. Soc.* **1995**, *117*, 7769–7775.
[9] S. T. Roweis and L.K. Saul, Nonlinear Dimensionality Reduction by Locally Linear Embedding, *Science* **2000**, *290*, 2323–2326.
[10] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1990.
[11] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. Van Der Vorst, Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide, Society for Industrial and Applied Mathematics, Philadelphia, 2000.
[12] O. Kouropteva, O. Okun, and M. Pietikainen, Selection of the Optimal Parameter Value for the Local Linear Embedding Algorithm, *Proceedings of the First International Conference on Fuzzy Systems and Knowledge Discovery*, Singapore, **2002**, pp.359–363.
[13] M. Vlachos, C. Domeniconi, D. Gunopulos, G. Kollios, and N. Koudas, Non–Linear Dimensionality Reduction Techniques for Classification and Visualization, in Proceedings of the 8th ACM SIGKDD *International Conference on Knowledge Discovery and Data Mining*, **2002**.
[14] C.G. Atkeson, A.W. Moore, and S. Schaal, Locally weighted learning, *A. I. Review* **1997**, *11*, 11–73.
[15] M. Birattari, G. Bontempi, and H. Bersini, Lazy learning meets the recursive least–squares algorithm, in: *Adv. in NIPS*, Eds. M.S. Kearns, S.A. Sola and D.A. Cohn, MIT Press, 1999. http://iridia.ulb.ac.be/~lazy/.
[16] R. H. Myers, *Classical and Modern Regression with Applications*, 2nd Edn., Boston, MA: PWS–KENT Publishing company, 1994.
[17] G. Bontempi, M. Birattari, and H. Bersini, A model selection approach for local learning, *A. I. Communications*

2000, *13*, 41–48.
[18] G. Bontempi, M. Birattari, and H. Bersini, Adaptive Memory Based Regression Methods, in: Proceedings of the 1998 IEEE World Congress on Computational Intelligence IJCNN' 98 (Anchorage, Alaska), 1997, pp. 2102–2106.
[19] W. S. Cleveland, S. J. Devlin, and E. Grosse, Regression by local fitting: Methods, properties, and computational algorithms, *J. of Econo*. **1988**, *37*, 87–114.
[20] O. Ivanciuc, Support Vector Machine Identification of the Aquatic Toxicity Mechanism of Organic Compounds, *Internet Electron. J. Mol. Des*. **2002**, *1*, 157–172, http://www.biochempress.com.
[21] O. Ivanciuc, Support Vector Machine Classification of the Carcinogenic Activity of Polycyclic Aromatic Hydrocarbons, *Internet Electron. J. Mol. Des*. **2002**, *1*, 203–218, http://www.biochempress.com.
[22] V. Vapnik, *Statistical Learning Theory*, Wiley, New York, 1998.
[23] C. J. C Burges, A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Disc*. **1998**, *2*, 121–167.
[24] N. Cristianini and J. Shawe–Taylor, *An introduction to Support Vector Machines*, Cambridge University Press, Cambridge, UK, 2000.
[25] S. Gunn, Support Vector Machines for Classification and Regression, ISIS Technical Report, **1997**.

Biographies

Rakesh Kumar is a project assistant in the chemical engineering division of National Chemical Laboratory, Pune, India. He obtained his bachelor's degree from Indian Institute of technology, Kharagpur.

Abhijit Kulkarni is a senior research fellow in the chemical engineering division of National Chemical Laboratory, Pune, India. His research interests include machine–learning applications in process engineering and process modeling, simulation and optimization. He obtained his bachelor's degree from University of Pune and Master's degree from Birla Institute of Technology and Science, Pilani. Presently he is doing PhD at University of Pune.

Valadi K. Jayaraman is a senior scientist in the chemical engineering division of the National Chemical Laboratory, Pune, India (jayaram@che.ncl.res.in). His interests include chemical and bio–reaction engineering, applications of artificial–intelligence tools in engineering, process modeling, optimization and control. Jayaraman has been visiting faculty to Indian universities and has taught many core chemical engineering courses to graduate students. He obtained his bachelor's and master's degrees in chemical engineering from the Univ. of Madras and his Ph.D. while working at National Chemical Laboratory. He has over 50 international publications. He has recently received the Herdillia award from the Indian Institute of Chemical Engineers (IIChE) for excellence in basic research.

Bhaskar D. Kulkarni is a senior scientist and heads the chemical engineering division of the National Chemical Laboratory (NCL), Pune, India. He has been with NCL for over 25 years. His interests are stochastic processes, non–linear systems, chemical reaction engineering, applications of artificial–intelligence tools in engineering, process modeling, optimization and control. A fellow of the Indian National Science Academy, National Academy of Sciences, National Academy of Engineering, and Third World Academy of Sciences, he has received numerous awards for his work. He has published three books and over 200 technical papers in prestigious international journals. Kulkarni obtained his bachelor 's and master 's degrees in chemical engineering from Laxminarayan Institute of Technology in Nagpur, India, and received his Ph.D. degree while working at NCL.