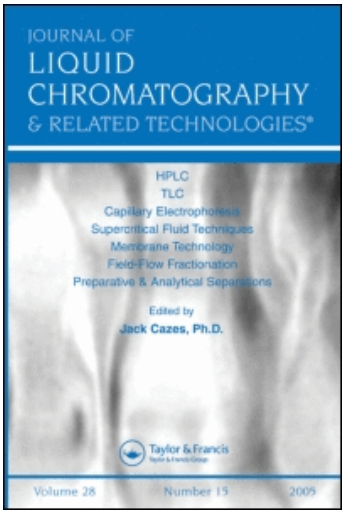


**Journal of Liquid Chromatography & Related Technologies**
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713597273

# RADIAL BASIS FUNCTION NETWORKS IN LIQUID CHROMATOGRAPHY: IMPROVED STRUCTURE-RETENTION RELATIONSHIPS COMPARED TO PRINCIPAL COMPONENTS REGRESSION (PCR) AND NONLINEAR PARTIAL LEAST SQUARES REGRESSION (PLS)

Yannis L. Loukas[a]
[a] Department of Pharmaceutical Chemistry, School of Pharmacy, University of Athens, Athens, Greece

Online publication date: 30 September 2001

**To cite this Article** Loukas, Yannis L.(2001) 'RADIAL BASIS FUNCTION NETWORKS IN LIQUID CHROMATOGRAPHY: IMPROVED STRUCTURE-RETENTION RELATIONSHIPS COMPARED TO PRINCIPAL COMPONENTS REGRESSION (PCR) AND NONLINEAR PARTIAL LEAST SQUARES REGRESSION (PLS)', Journal of Liquid Chromatography & Related Technologies, 24: 15, 2239 — 2256
**To link to this Article: DOI:** 10.1081/JLC-100105137
**URL:** http://dx.doi.org/10.1081/JLC-100105137

J. LIQ. CHROM. & REL. TECHNOL., 24(15), 2239–2256 (2001)

# RADIAL BASIS FUNCTION NETWORKS IN LIQUID CHROMATOGRAPHY: IMPROVED STRUCTURE-RETENTION RELATIONSHIPS COMPARED TO PRINCIPAL COMPONENTS REGRESSION (PCR) AND NONLINEAR PARTIAL LEAST SQUARES REGRESSION (PLS)

**Yannis L. Loukas**

Department of Pharmaceutical Chemistry, School of Pharmacy, University of Athens, Panepistimiopolis, Zografou, 157 71 Athens, Greece
E-mail: loukas@pharm.uoa.gr



## ABSTRACT

The application of the second most popular artificial neural networks (ANN), namely the radial basis function networks, has been developed for obtaining sufficient quantitative structure-retention relationships (QSRR) with improved accuracy. The present study examined a dataset of 25 substances as solutes to two different stationary phases (silica and alumina). The solutes were analyzed to their structural descriptors and related to their retention behavior, as expressed by their capacity factors, using radial basis function (RBF) and generalized regression neural networks (GRNN) as function approximation systems.

The proposed methods led to substantial gain in both the prediction ability and the computation speed of the resulting models

compared to regression models. Furthermore, the results were compared with that produced from classical linear and nonlinear multivariate regression such as principal components regression (PCR) and nonlinear (polynomial) partial least squares regression (PLS). Some of the proposed ANN models diminished the number of outliers, during their implementation to unseen data (solutes), to zero.

## INTRODUCTION

The interest in studies dealing with the retention behavior of solutes in different stationary phases has been increased substantially in the last years.[1] In these studies, the major aim is the development of improved expert systems capable to predict and describe the retention capability of different stationary phases, for the better understanding of retention process, and to provide a valuable chromatographic tool for highlighting into the molecular mechanisms of retention in a given HPLC system. Almost, in all of these studies, much effort is concentrated on the calculation of structural descriptors, which characterize the examined solutes. In literature, several models have been described, from linear to nonlinear ones, in order to calculate these values as accurate as possible.[2] In the present study, it is described for first time in the literature, the application of RBF and GRNN systems to predict accurately the capacity factor values of 25 structurally diverse aromatic solutes (Table 1) in two stationary phases in isocratic HPLC systems, without the prerequisite to specify any regression model (linear or nonlinear) and to provide efficient QSRR models.

QSRR studies could be used for both the selection of principal physicochemical characteristics (descriptors) and relating them to retention values, and the derivation of mathematical models that involve these multivariate data in order to be used for predictive purposes in every HPLC system. Multivariate data consist of the results of observations of many different variables (physicochemical descriptors) for a number of individuals (molecules). Each variable may be regarded as constituting a different dimension, such that, if there are $n$ variables each object may be said to reside at a unique position in an abstract entity referred to as $n$-dimensional hyperspace. This hyperspace is necessarily difficult to visualize, and the underlying theme of multivariate analysis (MVA) is, thus, the description of a polynomial in which the dependent variables are related to the independent variable(s). Known methods for this include the multiple regression analysis, experimental design techniques, nonlinear regression. The drawback, some times, of these very popular techniques is their inability to give highly predictive models due to hidden nonlinearity inside the data variables or the prerequisite to specify the mathematical model before the fitting of the data.

***Table 1.*** Structures, Physicochemical Parameters, and Observed Capacity Factors Values of the Examined Solutes in Two Stationary Phases

| | $R_2$ | $\pi_2^H$ | $\Sigma\alpha_2^H$ | $\Sigma\beta_2^H$ | $V_x$ | $\mu$ | $\delta_{max}$ | $V_{aq}$ | k′(1) | k′(2) |
|---|---|---|---|---|---|---|---|---|---|---|
| Hexylbenzene | 0.591 | 0.5 | 0 | 0.15 | 1.562 | 0.351 | 0.1326 | 649.7 | 4.56 | 3.721 |
| 1.3.5-Tris(1-methylethy)-benzene | 0.627 | 0.4 | 0 | 0.22 | 1.985 | 0.014 | 0.1309 | 777.15 | 4.887 | 4.147 |
| 1.4-Dinitrobenzene | 1.13 | 1.63 | 0 | 0.41 | 1.065 | 0 | 0.5652 | 451.02 | 0.969 | 0.774 |
| 3-(Trifluoromethyl)phenol | 0.425 | 0.87 | 0.72 | 0.09 | 0.969 | 2.096 | 0.4649 | 432.38 | 0.975 | 0.995 |
| 3.5-Dichlorophenol | 1.02 | 1.1 | 0.83 | 0 | 1.02 | 1.408 | 0.2239 | 440.2 | 1.502 | 1.528 |
| 4-Hydroxybenzonitrile | 0.94 | 1.63 | 0.79 | 0.29 | 0.93 | 3.313 | 0.2237 | 409.94 | 0.396 | 0.372 |
| 4-Iodophenol | 1.38 | 1.22 | 0.68 | 0.2 | 1.033 | 1.586 | 0.2213 | 434.24 | 1.174 | 1.173 |
| Methoxybenzene | 0.708 | 0.75 | 0 | 0.29 | 0.916 | 1.249 | 0.1481 | 407.98 | 0.835 | 0.589 |
| Benzamide | 0.99 | 1.5 | 0.49 | 0.67 | 0.973 | 3.583 | 0.3448 | 418.21 | 0.303 | −0.069 |
| Benzene | 0.61 | 0.52 | 0 | 0.14 | 0.716 | 0 | 0.1301 | 331.81 | 0.584 | 0.313 |
| Chlorobenzene | 0.718 | 0.65 | 0 | 0.07 | 0.839 | 1.307 | 0.1466 | 375.23 | 1.129 | 0.916 |
| Cyclohexanone | 0.403 | 0.86 | 0 | 0.56 | 0.861 | 2.972 | 0.1111 | 383.84 | 0.337 | 0.867 |
| Dibenzothiophene | 1.959 | 1.31 | 0 | 0.18 | 1.379 | 0.524 | 0.4465 | 555.67 | 3.041 | 3.126 |
| Phenol | 0.805 | 0.89 | 0.6 | 0.3 | 0.775 | 1.233 | 0.2173 | 353.02 | 0.099 | 0.047 |
| 1,1,2,3,4,4-Hexachloro-1,3-butadiene | 1.019 | 0.85 | 0 | 0 | 1.321 | 0.001 | 0.0606 | 516.73 | 3.248 | 3.426 |
| 1H-Indazole | 1.18 | 1.25 | 0.54 | 0.34 | 0.905 | 1.546 | 0.2752 | 405 | 0.822 | 0.647 |
| 3,7-Dihydro-1,3,7-trimethyl-1-H-purine-2,6-dione | 1.5 | 1.6 | 0 | 1.35 | 1.363 | 3.708 | 0.401 | 569.32 | 1.616 | 1.042 |
| 4-Nitrobenzoic acid | 0.99 | 1.07 | 0.62 | 0.54 | 1.106 | 3.431 | 0.5643 | 467.67 | −0.899 | −0.924 |
| 1-Methyl-2-pyrrolidone | 0.491 | 1.5 | 0 | 0.95 | 0.82 | 3.594 | 0.307 | 381.5 | 0.257 | −0.699 |
| Napthalene | 1.34 | 0.92 | 0 | 0.2 | 1.085 | 0 | 0.132 | 458.91 | 1.769 | 1.583 |
| 4-Chlorophenol | 0.915 | 1.08 | 0.67 | 0.2 | 0.898 | 1.478 | 0.2201 | 396.25 | 0.758 | 0.758 |
| Methylbenzene | 0.601 | 0.52 | 0 | 0.14 | 0.716 | 0.263 | 0.1301 | 384.44 | 1.027 | 0.829 |
| Piperazine | 0.57 | 0.83 | 0.2 | 1.17 | 0.763 | 1.995 | 0.1583 | 355.56 | 0.797 | 0.252 |
| Piperidine | 0.422 | 0.46 | 0.1 | 0.69 | 0.804 | 1.168 | 0.1554 | 368.5 | 0.574 | −0.021 |
| Benzonitrile | 0.742 | 1.11 | 0 | 0.33 | 0.871 | 3.335 | 0.1451 | 388.93 | 0.705 | 0.337 |

$R_2$: excess molar refraction; $\pi_2^H$: solute dipolarity/polarizability; $\Sigma\alpha_2^H$: solute overall hydrogen bond acidity; $\Sigma\beta_2^H$: solute overall hydrogen bond basicity; $V_x$: McGowan characteristic volume; $\mu$: total dipole moment; $\delta_{max}$: electron excess charge on an atom in solute molecule; $V_{aq}$: solvent (water) accessible molecular volume.

So, there is a need to improve further, such kind of models in order to extract the most accurate prediction. To this end, artificial neural networks (ANN),[3,4] and especially the "supervised" ones, could be used successfully in QSRR studies providing better results than the conventional regression models.

For a given data set, there are some patterns (e.g., a series of different descriptors) which have desired known responses (capacity factors). These two types of data (the representation of the objects and their responses in the system) form pairs, which for the present purpose are called inputs and targets. The goal of supervised learning is to find a *model* or *mapping* that will correctly associate the inputs with the targets. In the following, we shall examine the behavior of feedforward neural networks, such as RBF and GRNN and the results will be compared with that from the development of PCR and PLS methods.[4]



## EXPERIMENTAL

An extremely powerful type of feedforward artificial neural networks is the radial basis function (RBF) network, which differ strongly from the multilayer perceptron (MLP) network in the activation functions and how they are used. Generally, a network with three layers of weights and sigmoidal activation functions can approximate to arbitrary accuracy any smooth mapping. Furthermore, Bishop[5] appeals to the intuitive idea that any reasonable function can be approximated to arbitrary accuracy by a linear superposition of a sufficiently large number of localized 'bumb' functions. A two-layer network in which each hidden unit generates a bump-like function directly could achieve this. Such networks are called local basis networks, with most of our attention focusing on Gaussian basis functions since, as well as being localized they have a number of analytical properties.

The same conclusions are supported by the studies of Hartman[6] Park, and Sandberg.[7] Finally, Girosi and Poggio[8] have shown that radial basis function networks possess the property of *best approximation*, a property not shared by MLPs. To this end, in the present study, the RBF neural network architecture[9] was chosen to implement, for the first time, a QSRR study.

An RBF network (Figure 1a), therefore, has a hidden layer of radial units, each actually modeling a Gaussian response surface. Since these functions are non-linear, it is not actually necessary to have more than one hidden layer to model any shape of function: sufficient radial units will always be enough to model any function. The remaining question is how to combine the hidden radial unit outputs into the network outputs? It turns out to be quite sufficient to use a linear combination of these outputs (i.e., a weighted sum of the Gaussians). The RBF has an output layer containing linear units with linear activation function. The advantages and disadvantages of RBF over MLP networks are presented in details elsewhere.[10]

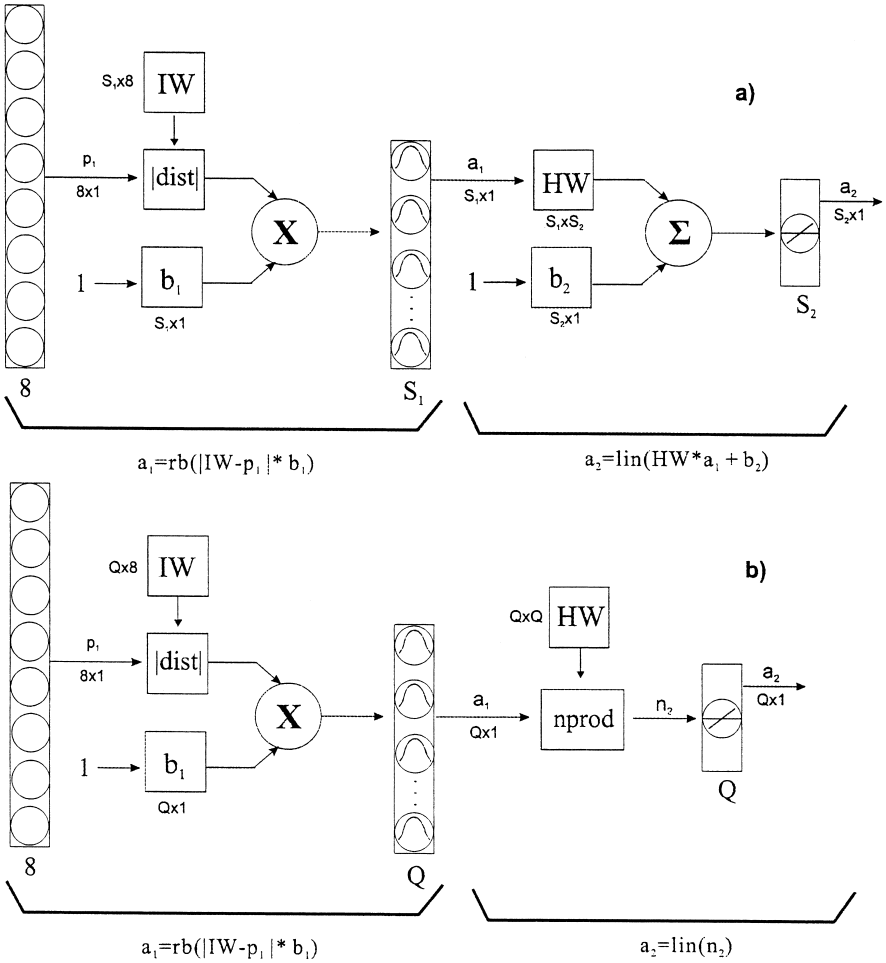


***Figure 1.*** Schematic representation of the architectures and the way of processing of the examined networks: 1a RBF and 1b GRNN.

Before linear optimization can be applied to the output layer of an RBF network, the number of radial units must be decided, and then their centers and deviations must be set. Centers should be assigned to reflect the natural clustering of the data. The two most common methods are: a) Sub-sampling: randomly-chosen training points are copied to the radial units. Since they are randomly selected, they will represent the distribution of the training data in a statistical sense. However, if the number of radial units is not large, the radial units may

actually be a poor representation.[11] b) K-Means algorithm: this algorithm[5] tries to select an optimal set of points, which are placed at the centroids of clusters of training data. Each training point belongs to a cluster center, and is nearer to this center than to any other center. Each cluster center is the centroid of the training points, which belong to it. If these algorithms fail to converge there are other algorithms (unsupervised or supervised) that should be examined.[5]

Once centers are assigned, deviations are set. The size of the deviation (also known as spread) determines how spiky the Gaussian functions are. Deviations should typically be chosen so that Gaussians overlap with a few nearby centers. Methods available are manually, heuristically or k-nearest neighbor. In the present study, manual selection of spread parameters (trial and error) was performed. Once centers and deviations have been set, the output layer can be optimized using the standard linear optimization technique: the pseudo-inverse (singular value decomposition) algorithm.[12]

One variant of RBF, which performs regression tasks, is the "General Regression Neural Network" GRNN[13] (Figure 1b), a term similar to kernel regression. It resembles a normalized RBF network in which there is a hidden unit centered at every training case. GRNN is a universal approximator for smooth functions, so it should be able to solve any smooth function-approximation problem, given enough data. The main drawback of GRNN is that, like kernel methods in general, it suffers badly from the curse of dimensionality. GRNN cannot ignore irrelevant inputs without major modifications to the basic algorithm. GRNNs have advantages and disadvantages: they can only be used for regression problems, train almost instantly, but tend to be large and slow (although it is not necessary to have one radial unit for each training case, the number still needs to be large), and like an RBF network, do not extrapolate.



## Multivariate Regression Analysis (Polynomial PLS and PCR)

The predictive ability of the examined ANN was compared further to that of classical multivariate regression. A popular technique for multivariate regression is the partial least squares (PLS) regression with cross-validation as an important concept that helps to identify the appropriate number of factors (or latent variables, *lv*) to use. Generally, PLS can be used to develop regression models that relate to a number of independent predictor variables (X-block) to one or more dependent or predicted variables (Y-block). PLS relies on a decomposition of the X-block (the physicochemical descriptors in the present study) based on covariance criteria. PLS finds factors (latent variables) that are descriptive of X-block variance and are correlated with the Y-block (capacity factors). PLS is advantageous to ordinary multiple linear regression (MLR) since it exam-

ines for collinearities in the predictor variables (i.e. some variables are linear combinations of other variables). The PLS models converges to MLR solution if all latent variables are included in the model.

There are several ways to calculate PLS models, with the most commonly used the non-iterative partial least squares (NIPALS) and the SIMPLS algorithms, both of them giving exactly the same results for univariate. The computational approaches for these two algorithms is well described in textbooks and it is beyond the scope of the present study. The polynomial PLS model works just like the linear PLS using the same algorithms, except once a pair of latent vectors is calculated, a polynomial of specific degree *n* is used to calculate the inner relation, replacing the *b* scalar for each latent variable with a *b* vector of polynomial coefficients. In the outputs of the function, *b* is a matrix (*n*+1 by *lv*). In the present study, it was confirmed that a polynomial with a degree of 2 generalizes better than the higher degrees polynomials.

Principal components regression (PCR) is a well-known and popular technique for forming regression models in systems where there is a good deal of variance in the independent or predictor variables. PCR works by doing a PCA decomposition of the predictor variables (X-block), then regressing the PCA scores against the predicted variable(s) (Y-block).



## RESULTS

We have performed a QSRR study using the data from literature[14] as presented in Table 1. For the development and evaluation of the artificial neural network (ANN) systems, the same (eight) descriptors proposed in literature were used in the present study also (see footnote of Table 1). The nonlinearity of the examined data set is highlighted in Figure 2, where a half matrix scatterplot summarize the correlations between the examined input variables. From Figure 2, it is evident there is a strong linear relationship between inputs 5 and 8 and one should decide to exclude one of these variables from the input layer. Since the purpose of the present study is the comparison with the published results (where inputs 5 and 8 are present), it was decided not to exclude one of these inputs from the architecture of RBF, PLS, and PCR models. This particular set of solutes has been studied already, and is ideal for the purpose of comparison.

The neural network systems were simulated using Matlab Neural Network Toolbox[15] running on a Pentium II platform. The input data were scaled before entering the RBF network for training. Training continued until there was no further decrease in overall error after a period of 1000 cycles and the average training time for each run was few minutes for the examined RBF networks. The eight inputs correspond to the eight descriptors and the one output to the logarith-

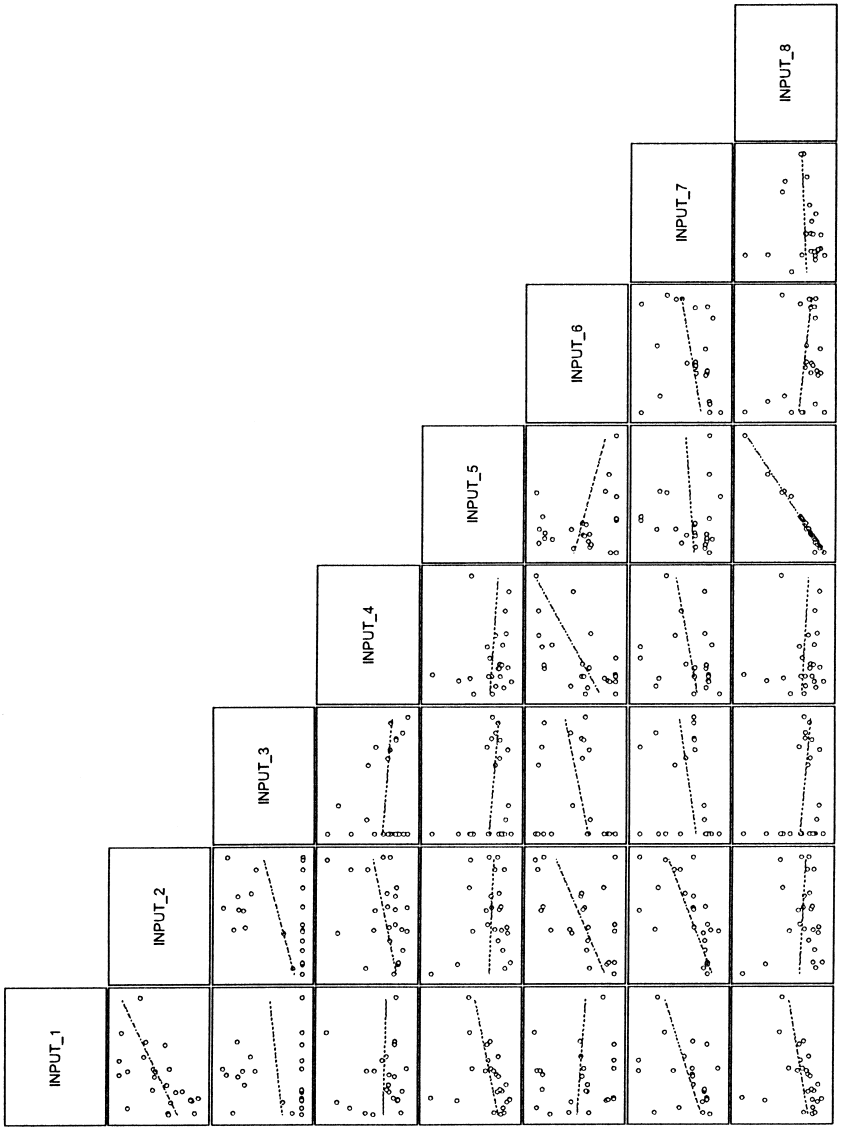


*Figure 2.* Half matrix scatterplot for the correlations between the examined input variables.

mic form of capacity factor values (Table 1). The quality of the examined RBF networks was assessed by two statistical variables: post training correlation coefficient and the number of outliers in the test subset (the meaning of outlier it is presented as footnote in Table 2).

## Implementation of RBF Networks

### Design of Training-Test Subsets

As far as the input variables and the dataset were selected, the next step was the division of the dataset in three subsets, namely the training and test subsets. The main requirement during training is the data presentation, meaning that the samples in the data set should be (evenly) spread over the expected range of data variability. In order to avoid the risk of representative samples which are not selected during training, we evaluated two different strategies of training set design as suggested from Massart,[16] namely the D-optimal design and the Kohonen self-organizing map approach.

Briefly, D-optimal designs are performed whenever the classical symmetrical designs cannot be used, because the experimental region is not regular in shape, or the number of experiments selected by a classical design is too large. The principle of this method is to select the experimental points to maximize the determinant of the information matrix $|X'X|$. This matrix is equal to the variance covariance matrix when X is defined as a matrix with n′ objects and m′ variables after centering (where n′ is the number of samples to be selected). The determinant of this matrix is maximal when the selected objects span the space of the whole data; in other words select the most influential samples (maximal spread). There are several algorithms with Fedorov iterative algorithm[17] which can be the best choices. We apply Fedorov's algorithm with the initial points selected by Sequential or Dykstra method,[18] starting with an empty design, searching through the candidate list of samples, and choosing, in each step, the one that maximizes the chosen criterion. There are no iterations involved; the requested number of points will simply be picked sequentially. The D-optimality method selects the samples for the linear model $y = \Sigma\, b_i x_i + e$, where $x_i$ = the input variable i. From this procedure, the samples 2, 3, 4, 6, 7, 13, 15, 17, 18, 19, 22, 23, 25, from Table 1 were chosen and were included in the training data set.

Next a clustering technique, the Kohonen network, was adopted to select the training cases. Simon et al[19] found that Kohonen self-organizing maps performed best in a similar task. The main goal of the Kohonen neural network is to map objects from n-dimensional into two-dimensional space. Objects with similar properties in the original space will map to the same node. In the present

study, a (3 x 3) Kohonen network was chosen containing 9 nodes. The learning rate was above 0.1 at the beginning and was linearly decreased to reach 0.01 at the end. The neighborhood size was also decreased linearly to reach a minimum of 1 after half of the training cycles and to remain 1 for the rest of the training. After stabilization of the network, it was observed that the samples chosen above were spread in all of the activated nodes, denoting again the representation of the whole space (Figure 3). Similarly, the representative list of the test sub set was isolated in the same manner.

## Training the Three RBF Networks

In the present study, two different RBF architectures were examined. In the first case, we start our examinations from the exact RBF (RBFE), which perform



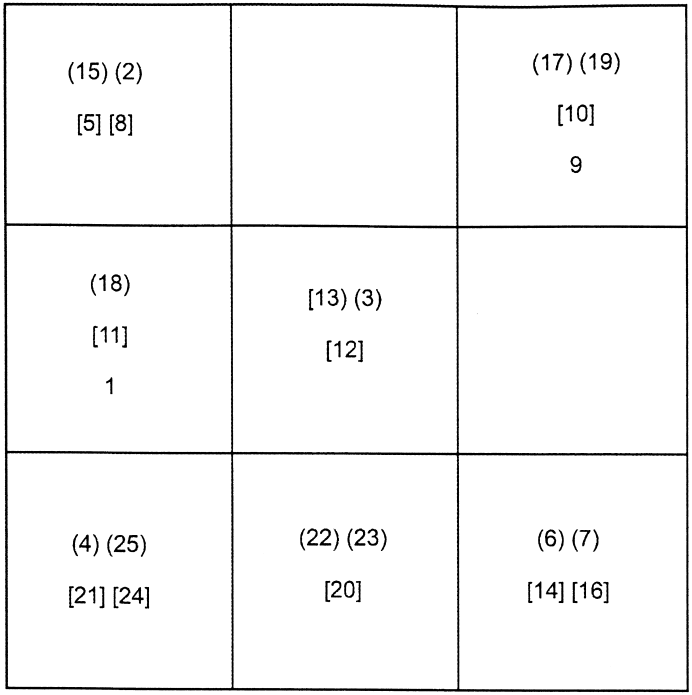


***Figure 3.*** Mapping of all 25 solutes into the 3 x 3 Kohonen map.

exact interpolation. Exact interpolation is a highly oscillatory function, which is generally undesirable and since the number of basis functions is equal to the number of patterns in the data set, for large data sets the mapping function can become very costly to evaluate. For the best generalization an interpolating function much smoother should be preferred. In RBFE architecture the network takes matrices of input vectors **p** and target vectors **t**, and a spread constant **spread** for the radial basis layer, and returns a network with weights and biases such that the outputs are exactly *t* when the inputs are *p*. This RBFE network creates as many *rb* (radial basis) neurons as there are input vectors in **p**, and sets the first layer weights to **p'**. Thus, we have a layer of *rb* neurons in which each neuron acts as a detector for a different input vector. If there are *Q* input vectors, then there will be *Q* neurons.

We have a QSFR problem with *C* constraints (input/target pairs) and each neuron has *C* +1 variables (the *C* weights from the *C rb* neurons, and a bias). A linear problem with *C* constraints and more than *C* variables has an infinite number of zero error solutions. Thus, RBFE creates a network with zero error on training vectors. The only condition we have to meet is to make sure that parameter **spread** is large enough so that the active input regions of the neurons overlap enough so that several neurons always have fairly large outputs at any given moment. This makes the network function smoother and results in better generalization for new input vectors occurring between input vectors used in the design. However, **spread** should not be so large that each neuron is effectively responding in the same, large, area of the input space. The drawback to RBFE is that it produces a network with as many hidden neurons as there are input vectors. For this reason, RBFE does not return an acceptable solution when many input vectors are needed to properly define a network, as is typically the case.

To overcome the problem of RBFE, we examine a more efficient network, the RBF, which iteratively creates a radial basis network adding one neuron at a time. Neurons are added to the network until the sum-squared error falls beneath an error goal or a maximum number of neurons have been reached. The RBF network takes matrices of input and target vectors, **p** and **t**, and design parameters **error goal** and **spread** and returns the desired network. The design method of RBF is similar to that of RBFE. The difference is that RBF creates neurons one at a time. At each iteration the input vector, which will result in lowering the network error the most, is used to create a *rb* neuron. The error of the new network is checked, and if it is low enough the RBF is stopped. Otherwise the next neuron is added. This procedure is repeated until the error goal is met, or the maximum number of neurons is reached. Usually, a third (validation) subset is used to indicate the end of training. In the present case, as in many other cases in literature,[10,16,20] this third subset is avoided and another criterion for stopping training and controlling the overfitting phenomenon is used. In the present study, the mean square training error MSETr was adopted as stopping criterion:

$$MSETr = \sum_{i=1}^{N}\sum_{j=1}^{g}\frac{(y_{ij} - out_{ij})^2}{Ng}$$

where N is the number of objects in the training data set, g is the number of output variables, $y_{ij}$ is the element of target matrix **y** (N x g) for the data considered (i.e training set) and $out_{ij}$ is the element of the output matrix **out** (N x g) of the RBF. In the present case, the RBF networks trained almost instantly so different error goals were examined for stopping training. The selection of the best RBF networks was based on their generalization ability (see later).

The second radial basis architecture examined is the GRNN. It is similar to the radial basis network, but has a slightly different second layer. In the case of GRNN network, we performed the same task as for the RBF networks above, in order to get a function that fits individual data points fairly closely.



## Generalization Procedure

The term generalization means the ability of the examined models to predict the outputs in unseen data (test data set). Although, the examined RBF networks were trained using the MSETr term as stopping criterion, in Table 4 is calculated the relative standard error for predictions (%RSEP); a statistical term for comparing the performance of the examined models (RBF, PLS, PCR and published results) in the same data set:

$$\%RSEP = \sqrt{\frac{\sum_{1}^{n}(y_{pred} - y_{obs})^2}{\sum_{1}^{n}y_{obs}^2}} \times 100$$

## RBF Networks

Having established each one of the examined RBF networks, a testing procedure was carried out. In this process, ten compounds were removed (selected in the unbiased way described above) from the complete data set prior to training and served as the test set. After training, the parameters of the ten compounds unknown to the network were put into the network and the predictive activities of these compounds were obtained.

With the correct weight and bias values for each layer and enough hidden neurons, a radial basis network can fit any function with any desired accuracy. It is important that the **spread** parameter be large enough that the *rb* neurons respond to overlapping regions of the input space, but not so large that all the neurons respond in essentially the same manner. If the spread of the radial basis neurons is too high, each neuron responds essentially the same and due to the large overlap of the input regions of the radial basis neurons the network cannot be designed. All the neurons always output 1 and so they cannot be used to generate different responses. To see how the network performs with the different **spread** values we perform a trial and error study as it appears in Table 2. From Table 2, it is becoming evident that as the **spread** increases from 0.1 to 50 the number of outliers drops from 6 to 2. Then it increases again to 5 with a spread value of 100. It is noticeable, also here, that the optimum **spread** value of 50 resulted in a correlation coefficient R-sq. of 0.883, much lower than the highest correlation coefficient R-sq. (0.968) resulted from a spread value of 0.1 with 6 outliers. This highlights the problem of over fitting, where the network trains perfect but generalize (predict new data) poorly.

The same procedure was followed also for the GRNN networks, where the **spread** value of 60 gave the best performance (three outliers in the test data set, Table 3). The larger the spread is, the smoother the function approximation is. A small spread value of 0.1 fit perfect the training data set (post training coefficient of 1) but generalizes poor (8 outliers). As spread value increases the post training

***Table 2.*** RBF Networks Trained with Different Spread Values, the Resulting Number of Outliers in the Test Subset as well as the Post Training Correlation Coefficient R-sq. in the Training Subset

| Spread | Post Training Correlation Coefficient R-sq. | Outliers[a] |
|---|---|---|
| 0.1 | 0.968 | 6 |
| 0.5 | 0.935 | 5 |
| 10 | 0.926 | 5 |
| 20 | 0.911 | 4 |
| 30 | 0.909 | 3 |
| 40 | 0.899 | 3 |
| 50 | 0.883 | 2 |
| 60 | 0.914 | 3 |
| 100 | 0.927 | 5 |

[a]An outlier had $|\text{binding}_{obs} - \text{binding}_{pred}| > 0.2$.

***Table 3.*** GRNN Networks Trained with Different Spread Values, the Resulting Number of Outliers in the Test Subset as well as the Post Training Correlation Coefficient R-sq. in the Training Subset

| Spread | Post Training Correlation Coefficient R-sq | Outliers |
|---|---|---|
| 0.1 | 1.000 | 8 |
| 1 | 0.981 | 5 |
| 10 | 0.905 | 4 |
| 50 | 0.883 | 4 |
| 60 | 0.875 | 3 |
| 70 | 0.867 | 3 |



coefficient decreases with equal increase in predictive performance (fit the data more smoothly). The RBF with **spread** 50 and the GRNN with **spread** 60 resulted in the best performance with RBF resulting in lower outliers and predictive values of higher accuracy, closer to the observed ones. Table 4 summarizes the %RSEP term, which characterizes the predictive abilities of the examined models. It is obvious that the superior performance of RBF network can predict the capacity factors of ten new (unseen) solutes to the examined stationary phases.

***Table 4.*** Test Subset of Ten Solutes, the Observed (Experimental) Capacity Factors, the Predicted from the RBF Network with Spread Value of 50 (see Table II), the Predicted from the PLS and PCR Methods and the Published Results. The Comparison of the Prediction Abilities of the Examined Models Was Based on Their %RSEP Values

| Solutes | Experimental | RBF Predicted | PLS Predicted | PCR Predicted | Published[14] |
|---|---|---|---|---|---|
| 5 | 1.502 (1.528)[a] | 1.487 (1.488) | 1.349 (1.285) | 1.040 (1.121) | 1.277 (1.039) |
| 8 | 0.835 (0.589) | 0.925 (0.765) | 1.084 (0.972) | 1.286 (1.014) | 1.206 (1.003) |
| 10 | 0.584 (0.313) | 0.755 (0.525) | 0.590 (0.324) | 0.976 (0.850) | 0.778 (0.707) |
| 11 | 1.129 (0.916) | 0.865 (0.989) | 0.765 (0.599) | 1.118 (0.941) | 0.859 (0.695) |
| 12 | 0.337 (0.867) | 0.465 (0.658) | 0.186 (0.107) | 0.580 (0.129) | 0.647 (0.416) |
| 14 | 0.099 (0.047) | 0.225 (0.135) | 0.533 (0.319) | 0.345 (0.325) | 0.440 (0.318) |
| 16 | 0.822 (0.647) | 0.745 (0.798) | 0.929 (0.808) | 0.624 (0.604) | 0.730 (0.540) |
| 20 | 1.769 (1.583) | 1.985 (1.854) | 1.735 (1.657) | 2.096 (1.958) | 2.082 (1.850) |
| 21 | 0.758 (0.758) | 0.695 (0.690) | 0.874 (0.746) | 0.629 (0.632) | 0.818 (0.633) |
| 24 | 0.574 (0.021) | 0.675 (0.242) | 0.712 (0.523) | 0.823 (0.433) | 0.798 (0.648) |
| %RSEP | — | 14.84 (18.98) | 19.54 (39.72) | 31.23 (44.88) | 26.76 (41.92) |

[a]Numbers in parenthesis correspond to capacity factors with the second stationary phase.

Quadratic PLS and PCR Models

Further to the RBF networks the same data set of the 25 solutes was used to perform classical multivariate regression such as PCR and quadratic PLS, which attempts to maximize covariance (to do both, capture variance and achieve correlation) and variance, respectively. The question that arises is: how many factors (latent variables) should be chosen? In the case of PCR, the first four components were used which captured 88.75% of the total variance with an eigenvalue of 0.807 and the lowest root-mean-square error of cross-validation (RMSECV), which is a measure of model's ability to predict new samples. RMSECV and RMSEC (root-mean-square error of calibration; tell us about the fit of the model to the calibration data) were represented graphically against a number of factors (Figure 4) where the conclusion for selecting four factors are supported. Adding the remaining factors, the RMSEC continues to decrease while there is a significant increase in RMSECV. Similarly, in the case of quadratic PLS, the model with 2 latent variables performs the best prediction. The development of a PLS model with 2 latent variables and a PCR model with the first four PCs resulted in the predictions shown in Table 4, together with the published results. From the %RSEP values, it is evident that the performance of PLS model is slightly better than the published results, while the worst predictions were derived from the PCR model.



## Sensitivity Analysis

In the present study, in order to draw some conclusions on the relative importance of the used input variables (the eight descriptors), we conducted a sensitivity analysis[21] on the inputs to the best neural network derived from the above analysis (see Table 2). This indicates which input variables are considered most important by that particular neural network. Sensitivity analysis rates variables according to the deterioration in modeling performance that occurs if that variable is no longer available to the model. Sensitivity analysis does not rate the "usefulness" of variables in modeling in a reliable or absolute manner. It simply indicates the performance of the network if that variable is "unavailable" (important variables have a high error, indicating that the network performance deteriorates badly if they are not present).

In the present study, the importance of the inputs was the following: the most influential descriptors are those related to the solute volumes ($V_{aq}$ and $V_x$ - see footnote of Table 1). Even though there is a strong relationship ($r_{corr} = 0.99$) between these two volume variables and one of them should be discarded as redundant, we retained both in the design of the examined models since we used the descriptors from literature without any preprocessing (variable selection) pro-

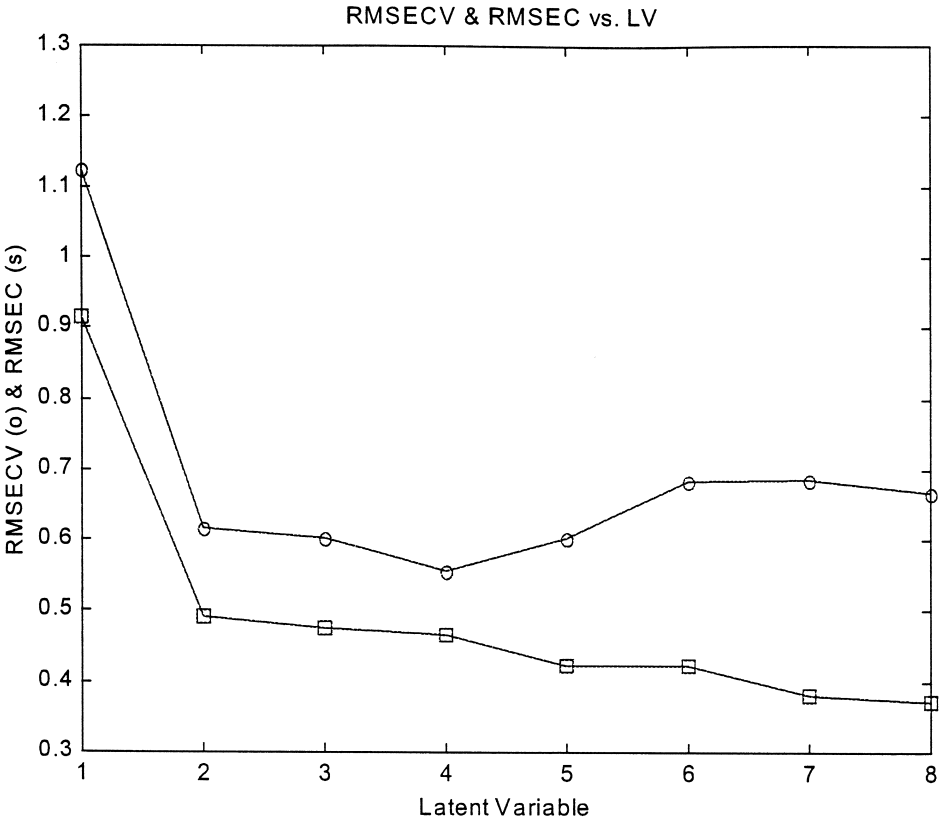


***Figure 4.*** RMSEC and RMSECV vs. latent variables in PCR modeling procedure.

cedure. The next important variables are the ones related to the charge conditions and polarity of the solutes (dipolarity $\pi_2^H$ and dipole moment $\mu$). Next to the size and polarity of molecules the variables related to the ability of solutes to form hydrogen bonds with the molecules of either the aqueous or stationary phase ($\Sigma\alpha_2^H$ and $\Sigma\beta_2^H$) follow in importance. The current observations highlight the mechanism of migration procedure through the aqueous phase for the specific stationary phases. Using a different stationary phase the prediction of the chromatographic elution will be possibly based on different descriptors highlighting a different chromatographic mechanism. The selection of the most important descriptors could be of great interest to the research dealing with chromatographic separations.

## CONCLUSION

The data set of the 25 diverse substances as solutes to different stationary phases, is a representative sample from the population of solute:stationary phase interactions, where it is necessary to model the interaction procedure and to predict the capacity factor values. In the present study, the examined radial basis function networks, namely RBF and GRNN, as function approximate systems, behaved with high accuracy and outperformed linear and nonlinear multiple regression systems. If we add to the increased accuracy of RBF networks, the lack of difficulty to find an optimum architecture and the almost instant training, it could be easily concluded that RBFs could be a significant partner to the development of different QSRR systems.



## REFERENCES


1. Balcan, M.; Cserhati, T.; Forgacs, E.; Anghel, D.F. Biomed. Chromatogr. **1999**, *13*, 225.
2. Forgacs, E.; Cserhati, T. J. Pharm. Biomed. Anal. **1998**, *18*, 505.
3. Zupan, J.; Gasteiger, J. *Neural Networks for Chemists*; VCH Verlagsgesellschaft mbH: Weinheim, 1993.
4. *Handbook of Chemometrics and Qualimetrics*: Parts A and B; Massart, D.L.; Vandeginste, B.G.M.; Buydens, L.M.C.; de Jong, S.; Lewi, P.J.; Smeyers-Verbeke, J., Eds., Elsevier Science B.V.: Amsterdam, 1998.
5. Bishop, C. *Neural Networks for Pattern Recognition*; Oxford: University Press: 1995.
6. Hartman, E.J.; Keeler, J.D.; Kowalski, J.M. Neur. Comput. **1990**, *2*, 210.
7. Park, J.; Sandber, I.W. Neur. Comput. **1993**, *5*, 305.
8. Girosi, F.; Poggio, P. Biolog. Cybern. **1990**, *63*, 169.
9. Chen, S.; Cowan, C.F.N.; Grant, P.M. IEEE Trans.Neur.Networks **1991**, *2*, 302.
10. Loukas, Y.L. Anal. Chim. Acta **2000**, *in press*.
11. Haykin, S. *Neural Networks: A Comprehensive Foundation*; Macmillan Publishing: New York, 1994.
12. Golub, G.; Kahan, W. Numer. Anal. **1965**, *2*, 205.
13. Wasserman, P.D. *Advanced Methods in Neural Computing*; Van Nostrand Reinhold: New York, 1993.
14. Cserhati, T.; Forgacs, E.; Payer, K.; Haber, P.; Kaliszan, R.; Nasal, A. LC-GC-Int. **1998**, *11*, 240.
15. *Matlab Version 5.2*, MathWorks Inc.: Natick, MA.
16. Wu, W.; Walczak, B.; Massart, D.L.; Heuerding, S.; Erni, E.; Last, I.R.; Prebble, K.A. Chem. Intell. Lab. Syst. **1996**, 33, 35.

17. Fedorov, V.V. *Theory of Optimal Experiments*, Academic Press: New York, 1972.
18. Dykstra, O. Jr. Technometrics **1971**, *13*, 682.
19. Simon, V.; Gasteiger, J.; Zupan, J. J. Am. Chem. Soc. **1993**, *115*, 9148.
20. Majcen, N.; Kanduc, K.R.; Novic, M.; Zupan, J. Anal. Chem. **1995**, *67*, 2154.
21. Despagne, F.; Massart, D.L. Analyst **1998**, *123*, 157R-178R.

Received October 20, 2000 Manuscript 5432
Accepted January 2, 2001