GRADE INTERPOLATION USING RADIAL BASIS FUNCTION NETWORKS

I. K. Kapageridis
kapa1@hol.gr; http://users.hol.gr/~kapa1/;Department of Geotechnology and Environmental Engineering, Technological Education Institute of West Macedonia, Greece

ABSTRACT: This paper analyses the application of Radial Basis Function (RBF) networks in grade interpolation. These networks are a very unique member of the family of Artificial Neural Networks. RBF networks have such theoretical properties that establish them as a potential alternative to existing grade interpolation techniques. Their suitability to the problem of grade interpolation will be demonstrated in this paper both theoretically and through a number of case studies from real and simulated mineral deposits.

1 INTRODUCTION

This paper gives an analysis of a very unique type of Artificial Neural Networks, the *Radial Basis Function* networks (RBF) and their suitability to the problem of grade interpolation. RBFs were initially used for solving problems of real multivariate interpolation. Work on this subject has been extensively surveyed by Powell (Powel 1981). The theory of RBFs is one of the main fields of study in numerical analysis (Powel 1992, Singh 1992).

RBF networks are very simple structures. Their design is in essence a problem of curve fitting in a high-dimensional space. Learning in RBF networks means finding the hyper-surface in multi-dimensional space that fits the training data in the best possible way. Function approximation and pattern classification are the main areas of RBF networks application. One of the main advantages of RBF networks lies in their strong scientific foundation. RBFs have been motivated by statistical pattern processing theory, regression and regularisation, biological pattern formation, and mapping in the presence of noisy data (Powel 1992). Therefore, RBF networks have inherited a wide range of useful theoretical properties, which have been used to provide solutions to a much wider range of problems than the RBFs themselves.

2 THEORETICAL FOUNDATION

The basic principles of Radial Basis Functions and of the derived networks will be discussed in this section. For the purposes of this paper, the discussion will concentrate to the theory behind the use of RBFs for interpolation problems and not for pattern classification. The transition from the original RBF methods for interpolation to RBF networks will also be analysed.

2.1 Multivariable Interpolation

RBFs were first introduced to the problem of multivariable interpolation as an approach to dealing with irregularly positioned data points. The problem of multivariable interpolation is as follows (Powel 1981):

Given m different points $\{\underline{x}_i; i = 1,2,...,m\}$ in $\Re^n$, and m real numbers $\{f_i; i = 1,2,...,m\}$, one has to calculate a function s from $\Re^n$ to $\Re$ that satisfies the interpolation conditions

$$s(\underline{x}_i) = f_i, i = 1,2,...,m.$$

The choice of s from a linear space that depends on the positions of the data points forms the approach of radial basis functions. RBFs have the general form:

$$\phi(\|\underline{x} - \underline{x}_i\|), \underline{x} \in \Re^n, i = 1,2,...,m$$

Where ϕ is the basis function from $\Re^+$ to $\Re$ and the norm of $\Re^n$ is Euclidean. Several interpolation methods have been considered in which *s* has the form:

$$s(\underline{x}) = \sum_{i=1}^{m} \lambda_i \phi(\|\underline{x} - \underline{x}_i\|), \underline{x} \in \Re^n.$$

With the condition of the matrix

$$A_{ij} = \phi(\|\underline{x}_i - \underline{x}_j\|), i, j = 1,2,...,m,$$

being non-singular, the interpolation condition above defines the coefficients $\{\lambda_i; i = 1,2,...,m\}$ uniquely. The matrix *A* is normally called the *interpolation matrix*. These methods have a very useful property, proved by Micchelli (1986), that, if the data points are different then, for all positive integers *m* and *n*, *A* is always non-singular. This theory applies to many choices of ϕ. However, in the case of the basis functions of the form

$$\phi(r) = r^l, r \geq 0$$

Multiquadratics:

$$\phi(r) = \left(r^2 + c^2\right)^{1/2} \text{ for some c > 0 and } r \in \Re$$

Inverse Multiquadratics:

$$\phi(r) = \frac{1}{\left(r^2 + c^2\right)^{1/2}} \text{ for some c > 0 and } r \in \Re$$

Gaussian Functions*:*

$$\phi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \text{ for some } \sigma > 0 \text{ and } r \in \Re$$

Thin Plate Splines:

$$\phi(r) = r^2 \ln(r),\ r \in \Re$$

It should be noticed that multiquadratics and thin plate splines decrease by moving away from the centre of the basis function, while Gaussian and inverse multiquadratics increase. The thin plate splines are interpolating functions derived by variational methods (Duchon 1977, Meinguet 1979).

2.2 The Hyper-Surface Reconstruction Problem

The interpolation technique described above suffers from a very serious problem: If the number of data points in the training sample is greater than the number of degrees of freedom of the underlying physical process, then fitting as many RBFs as the number of data points leads to over-determination of the hyper-surface reconstruction problem (Broomhead and Lowe 1988). This is known in neural network terms as *overfitting* or *overtraining*. Allowing an RBF network to reach this stage means degradation of its generalisation performance.

The problem of learning the hyper-surface defining the output in terms of the input can be either *well-posed* or *ill-posed.* These terms have been in use in applied mathematics for over a century. An unknown mapping *f* between a domain *X* and an output range *Y* (both taken as metric spaces) is considered. Reconstructing this mapping *f* is said to be well-posed when the following three conditions are satisfied (Tikhonov and Arsenin 1977, Morozov 1993, and Kirsch 1996):

Existence: for every input vector $x \in X$ there is an output $y = f(x)$, where $y \in Y$.

Uniqueness: for any pair of input vectors $x, t \in X$, $f(x) = f(t)$ only if $x = t$.

Continuity: also referred to as *stability*, continuity requires for any $\varepsilon > 0$ there will exist $\delta = \delta(\varepsilon)$ so that if $\rho_x(x, t) < \delta$ then $\rho_y(f(x),f(t)) < \varepsilon$, where $\rho(\cdot,\cdot)$ is the distance between two arguments in their respective spaces (Haykin 1999).

A problem is ill-posed when any of these conditions is not satisfied. Normally, a physical phenomenon such as an orebody deposition, is a well-posed problem. Learning from drillhole data is, however, an ill-posed problem because (Kapageridis 1999):

- For any pair of input vectors *x*, *t* there can be $f(x) = f(t)$ even when $x \neq t$.
- It is well known that drillhole and other physical samples from mineral deposits contain large amounts of noise and imprecision leading to the possibility for the neural network to produce an output outside the range *Y* for a specified input. That means violation of the continuity criterion.

The second of the reasons has a more serious impact to solving the problem, as lack of continuity means that the computed input-output mapping does not represent the true solution.

The issues of hyper-surface reconstruction with RBFs being an ill-posed problem and leading to overfitting need to be addressed. A number of methods have been developed for making an ill-posed problem into a well-posed one, as well as preventing overfitting. The most important one, *regularisation*, will be discussed in the following paragraph.

2.3 Regularisation

Regularisation is a method developed by Tikhonov (1963) for solving ill-posed problems. Its use has been mostly explored in approximation theory. Regularisation aims at overcoming the lack of continuity of an ill-posed problem by means of an auxiliary nonnegative functional embedding prior information about the solution. Such information is commonly the assumption that similar inputs correspond to similar outputs. Tikhonov's theory involves two terms:

Standard Error Term: denoted by $E(F)$, represents the standard error or distance between the desired response (target output) d_i and the actual response y_i for the training example $i = 1, 2, \ldots, N$. The standard error term is defined as:

$$E_s(F) = \frac{1}{2}\sum_{i=1}^{N}(d_i - y_i)^2 = \frac{1}{2}\sum_{i=1}^{N}[d_i - F(x_i)]^2$$

Regularising Term: denoted by $E_c(F)$, provides the means for embedding geometrical information about the approximating function $F(x)$ to the solution. This term is defined as:

$$E_c(F) = \frac{1}{2}\|\mathbf{D}F\|^2$$

where **D** is a *linear differential operator*. It is in this operator that prior information about the form of the solution is embedded and therefore its selection depends on the problem at hand.

Regularisation provides a way of reducing the number of basis functions when fitting RBFs by adding a penalty term described above as the regularising term (Ripley 1995). The principle of regularisation is the following:

Find the function $F\lambda(x)$ that minimises the Tikhonov functional $E(F)$, defined by

$$E(F) = E_s(F) + \lambda E_c(F)$$

Where λ is a positive real number called the *regularisation parameter*. The choice of λ is very crucial as it controls the balance of contribution from the sample data and the prior information. It can also be seen as an indicator of the sufficiency of the given data samples to specify the solution to the above minimisation problem.

The implementation of the regularisation theory leads to the *regularisation network* (Poggio and Girosi 1990). As shown in Fig. 1, it consists of three layers. The first layer consists of a number of input nodes equal to the dimension m_o of the input vector x. The second or hidden layer consists of non-linear nodes connected directly to all the input nodes. The number of hidden nodes equals the number of samples N.

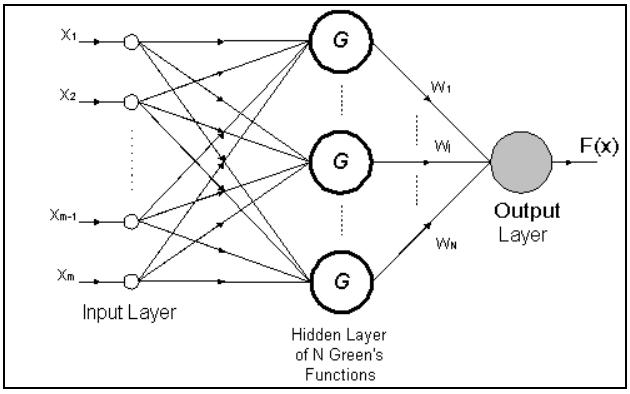

Figure 1: Regularisation network (Haykin 1999).

The activation function used in the hidden nodes is a *Green's function* $G(x, x_i)$. One of the most common Green's functions is the *multivariate Gaussian function:*

$$G(x, x_i) = \exp(-\frac{1}{2\sigma_i^2}\|x - x_i\|^2)$$

where x_i denotes the *centre* of the function, σ_i its *width* or *receptive field*, and w_j the unknown coefficients. These coefficients are defined as follows:

$$w_i = \frac{1}{\lambda}[d_i - F(x_i)], i = 1,2,..., N$$

The minimising solution, denoted as $F\lambda(x)$, is given by:

$$F_\lambda(x) = \sum_{i=1}^{N} w_i G(x, x_i)$$

The solution reached by the regularisation network exists in an N-dimensional subspace of the space of smooth functions, the set of Green's functions constituting the basis for this subspace (Poggio and Girosi 1990). As Poggio and Girosi point out, the regularisation network has three useful properties:

- It is a *universal approximator* as it can approximate arbitrarily well any multivariate continuous function, given sufficient number of hidden nodes.
- It has the *best-approximation property*, i.e. given an unknown non-linear function *f*, there always exists a choice of coefficients that approximate *f* better than all other choices.
- It provides the *optimal solution*. In other words, the regularisation network minimises a functional that measures the solution's deviation from its true value as represented by the training data.

3 RADIAL BASIS FUNCTION NETWORKS

The structure described above as the regularisation network has a very important weakness: as the number of functions depends initially to the number of training samples, the network produced can be very expensive in computational terms. This can be easily understood by considering the computation of the network's linear weights, which requires inversion of a very large matrix. Therefore there is a need for reducing the complexity of the network leading to an approximation of the regularised solution.

This is achieved by the introduction of a simplified version of the regularisation network, *the generalised radial basis function network*. From this point on, it will be assumed that RBF networks are generalised RBF networks. RBF networks involve searching for a sub-optimal solution in a lower-dimensional space. This solution approximates the regularised solution discussed before.

3.1 RBF Structure

Figure 2 illustrates the basic structure of the (generalised) RBF network. The first obvious difference between this network and that of Fig. 1 is in the number of hidden layer basis functions. In the RBF network there are m_1 RBFs, typically less than the number of training samples, while in the regularisation network there were N RBFs, with N equal to the number of training samples. Other structural differences include the number of weights being also reduced to m_1, and the introduction of a bias applied to the output unit.

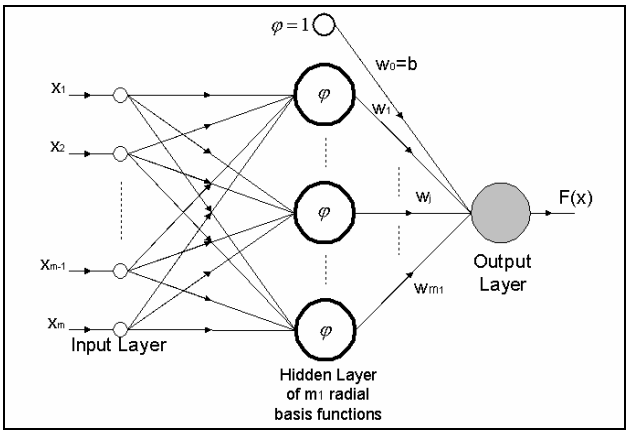

Figure 2: Structure of generalised RBF network.

Significant differences, not so obvious from the figures, concern the centre positions and receptive fields of the RBFs as well as the linear weights associated with the output layer. These are all unknown parameters and have to be learned by the RBF network during training. In the regularisation network, only the linear weights are unknown and require training. In the next paragraph, the function of the RBF network will be further analysed. Special attention is given to the way of initially positioning the RBF centres during initialisation and the RBF learning algorithms.

3.2 RBF Initialisation and Learning

For an RBF network to be able to receive training samples and function as a hyper-surface reconstruction network, a number of its parameters need to be calculated. These parameters include:

- The linear weights between hidden and output layer.
- The bias to the output units.
- The centres of the hidden layer RBFs.

There are a number of methods for RBF network initialisation and learning. The most common methods are:

Random Centre Selection: it is the simplest of the methods. The centres are randomly chosen from the training data set. It is a common method used when the training data represent well the problem at hand. Learning using this approach is concentrated in adjusting the linear weights between the hidden and output layer. This is achieved using the *pseudoinverse method* (Broomhead and Lowe 1988). The weights are calculated using the formula below:

$$w = G^{+} d$$

where d represents the target output vector in the training data set. $G+$ is the pseudoinverse of matrix G, defined as

$$G = \{g_{i,j}\}$$

where $g_{i,j}$ is the output of RBF i when presented with input vector j. Golub and Van Loan (1996) provide an in depth discussion over the computation of a pseudoinverse matrix.

Self-Organised Centre Selection: The learning method described above requires a data set representative to the problem at hand. Randomly selected centres do not necessarily reflect accurately the distribution of the data points. To overcome this problem, a clustering algorithm is used that creates homogeneous groups of data from the given data set. There are a number of clustering algorithms, however, in the case of RBF networks, the *k-means clustering algorithm* is the most commonly used (Duda and Hart 1973). Moody and Darken (1989) describe the use of *k*-means clustering algorithm. The number of centres k is set in advance. With the number of centres set, the algorithm proceeds with the following steps (Bishop 1995):

I. The values of the initial RBF centres $t_k(0)$ are set randomly. These values need to be different between them.

II. A vector x is selected from the data set and passed to the algorithm. The index $k(x)$ of the best-matching centre for the vector is calculated using the minimum-distance Euclidean criterion:

$$k(x) = \arg\min_{k} \|x(n) - t_k(n)\|, k = 1,2,...,m_1$$

where $t_k(n)$ is the kth centre at iteration n.

III. The RBF centres are adjusted using the following rule:

$$t_k(n+1) = \begin{cases} t_k(n) + \eta[x(n) - t_k(n)], k = k(x) \\ t_k(n), otherwise \end{cases}$$

where η is the *learning-rate parameter* receiving values between 0 and 1. This parameter controls the speed of learning, i.e. the degree of adjustment on the particular network parameter, in this case, the RBF centres.

IV. The iteration pointer n is increased by 1 and the algorithm loops back to step II. This process continues until the centres become stable.

The self-organised stage described above is followed by a supervised learning stage, which allows the calculation of the linear weights between the hidden and output layer. The overall approach depends largely on the initial selection of centres. Several enhancements to the initial centre selection have been introduced in order to avoid the situation where some initial centres get trapped in regions of the input space with low density of data points (Chen 1995, Chinunrueng and Sequin 1994). This learning method is used in the development stages of RBF networks for the problem of grade interpolation.

Orthogonal Least Squares: the OLS algorithm involves sequential addition of new RBFs to a network, which starts with a single basis function. Each new RBF is positioned to each data point and the linear weights are calculated for each position. The centre that gives the smallest residual error is retained. This way the number of RBFs increases step by step. The selection of a candidate data point for centre positioning is done by constructing a set of orthogonal vectors in the space *S* spanned by the hidden unit activation vectors for each training pattern. The data point that produces the greatest reduction in the residual error is chosen as the location of the new RBF centre. It is important to stop the algorithm well before every data point is selected to ensure good generalisation.

Supervised Centre Selection: the basis of this method is the *least-mean-square* algorithm (LMS). A supervised learning process based on the LMS algorithm sets all the free parameters of the RBF network. The LMS algorithm takes the form of a gradient descent procedure. Initially, a cost function is defined as follows:

$$E = \frac{1}{2}\sum_{j=1}^{N} e_j^2$$

where N is the number of training samples, and e_j is the error defined as:

$$\begin{aligned} e_j &= d_j - F^*(x_j) \\ &= d_j - \sum_{i=1}^{M} w_i G(\left\| x_j - t_i \right\|_{C_i}) \end{aligned}$$

where C_i is the norm-weighting matrix. The method aims at minimising E by adjusting the free parameters of the network, the weights w_i, the centres t_i, and the receptive fields Σ_i^{-1}. The adjustments to these three parameters are calculated below (Haykin 1999):

Linear Weights Adjustment:

$$\frac{\partial E(n)}{\partial w_i(n)} = \sum_{j=1}^{N} e_j(n) G(\left\| x_j - t_i(n) \right\|_{C_i})$$

Centres Position Adjustment:

$$\frac{\partial E(n)}{\partial t_i(n)} = 2w_i(n)\sum_{j=1}^{N} e_j G'(\left\| x_j - t_i(n) \right\|_{C_i}) \Sigma_i^{-1}[x_j - t_i(n)]$$

Receptive Fields Adjustment:

$$\frac{\partial E(n)}{\partial \Sigma_i^{-1}(n)} = -w_i(n)\sum_{j=1}^{N} e_j(n) G'\left(\left\| x_j - t_i(n) \right\|_{C_i}\right) Q_{ji}(n)$$

$$Q_{ji}(n) = [x_j - t_i(n)][x_j - t_i(n)]^T$$

The update rules for the three parameters, based on the three learning-rate parameters η_1, η_2, and η_3, are given below:

Linear Weights Update Rule:

$$w_i(n+1) = w_i(n) - \eta_1 \frac{\partial E(n)}{\partial w_i(n)}$$

Centres Positions Update Rule:

$$t_i(n+1) = t_i(n) - \eta_2 \frac{\partial E(n)}{\partial t_i(n)}$$

Receptive Fields Update Rule:

$$\Sigma_i^{-1}(n+1) = \Sigma_i^{-1}(n) - \eta_3 \frac{\partial E(n)}{\partial \Sigma_i^{-1}(n)}$$

It should be noticed that this gradient-descent procedure for RBF networks does not involve error back-propagation.

Regularisation Based Learning: the final RBF learning method described is based on regularisation theory. Yee (1998) provides the justification for this RBF design procedure that is based on four main elements:

A radial-basis function, *G*, admissible as the kernel of a mean-square consistent Nadaraya-Watson regression estimate (NWRE) (Nadaraya 1964, Watson 1964).

A common for all centres, input norm-weighting matrix, Σ^{-1}, with entries

$$\Sigma = diag(h_1, h_2, ..., h_{m_0})$$

where h_1, h_2, …, h_{mo} are the bandwidths of a consistent NWRE kernel G for each dimension of the input space. These bandwidths are given as the product of the sample variance of the ith input variable estimated from the available training data and a scale factor determined using a cross-validation procedure.

Regularised strict interpolation for the training of the linear weights using the following equation:

$$\mathbf{w} = (\mathbf{G} + \lambda \mathbf{I})^{-1}\mathbf{d}$$

where $\boldsymbol{G}$ is Green's matrix and $\boldsymbol{I}$ is the *N-by-N* identity matrix.

The choice of the regularisation parameter λ and the scale factors is achieved using a method such as the common cross-validation (CV). Generally, larger values of λ lead to larger

noise in measuring the parameters. In a similar manner, the larger the values for a specific scale factor, the less important is the associated input dimension for the variation of the network output in relation to variations in the input. In other words, the scale factors can be used for ranking the significance of the input variables and can aid the reduction of the input space dimensionality.

4 FUNCTION APPROXIMATION WITH RBF NETWORKS

In this section, the discussion continues with an evaluation of the function approximation capabilities of RBF networks. It will be shown that the range of RBF networks is broad enough to uniformly approximate any continuous function. The effects of the input space dimension and the amount of input data on the RBF network approximation properties will also be analysed.

4.1 Universal Approximation

The universal approximation theorem for RBF networks, as stated by Park and Sandberg (Park and Sandberg 1991), opened the way for their use in function approximation problems, which were commonly approached using Multi-Layered Perceptrons. The work of Park and Sandberg (1991, 1993), Cybenko (1989), and Poggio and Girosi (1990) led to a new model for function approximation based on generalised RBF networks. Specifically, the theorem can be stated as below:

Let $G:R^{mo}\rightarrow R$ is an integrable bounded function such that G is continuous and

$$\int_{R^{m_0}} G(x)dx \neq 0$$

Let $\Im_G$ denote the family of RBF networks consisting of functions $F:R^{mo}\rightarrow R$ represented by

$$F(x) = \sum_{i=1}^{m_1} w_i G\left(\frac{x - t_i}{\sigma}\right)$$

where $\sigma > 0$, $w_i \in R$ and $t_i \in R^{mo}$ for $i = 1, 2, \ldots, m_1$. For any continuous input-output mapping function f(x) there is an RBF network with a set of centres $\{t_i\}_{i=1}^{m_1}$ and a common receptive field $\sigma > 0$ such that the input-output mapping function F(x) realised by the RBF network is close to f(x) in the L_p norm, $p \in [1,\infty]$.

The universal approximation theorem provides the theoretical basis for the design of RBF networks for practical applications.

4.2 Input Dimensionality

A very critical issue in the use of RBF networks as function approximators is the dimension of the input space and its effect on the intrinsic complexity of the approximating function(s). It is generally accepted that this complexity increases exponentially in the ratio m_o/s, where m_o is the *input dimensionality* and s is a *smoothness index* of the number of constraints imposed on the approximating function. Therefore, for the RBF network to be able to achieve a sensible rate of convergence, the smoothness index s needs to be increased with the number of parameters in the approximating function. However, the space of approximating functions attainable with RBF networks becomes increasingly constrained as the input dimensionality is increased (Haykin 1999).

Increased dimensionality also has a great effect on the computational overhead caused during training of the RBF network. The dimension of the input space has a direct control over the RBF network architecture – the number of input nodes, the number of RBFs, and consequently, the number of linear weights between hidden and output layer. Therefore, any increase in the input dimensionality causes an increase in computer memory and power requirements, and an almost certain increase in development time. The most common ways of addressing the high input dimensionality for a given problem are to identify and ignore the inputs that do not contribute considerably to the output or to try to combine inputs that present a high correlation. Another way of reducing the input dimensionality, which is not always applicable though, is to try and break a complex problem into a number of low dimensionality problems that can be more effectively addressed using RBF networks.

4.3 Comparison of RBF Networks with Multi-Layer Perceptrons

Comparison of RBF networks with MLPs is inevitable since they are both used for similar applications and both are universal approximators. This comparison also leads to better understanding of these two ANN architectures. The differences between the two architectures are both structural (concerning the topology of the network) and functional (concerning the operation and use of the network):

Structural Differences:

- RBF networks have a single hidden layer. MLPs can have more than one hidden layers.
- Hidden units in RBF networks are different from the output units. MLP hidden units are similar to the output units.

Functional Differences:

- RBF networks construct local approximations to non-linear input-output mappings, while MLPs construct global approximations.
- The output layer of an RBF network is always linear, while the MLP output layer can be non-linear depending on the application.
- RBF hidden units calculate the Euclidean norm between the input vector and their centre, while

MLP hidden units compute the inner product of the input vector and their synaptic weight vector.

- MLPs exploit the logistic non-linearity to create combinations of hyperplanes to dissect pattern space into separable regions. RBF networks dissect pattern space by modelling clusters of data directly and, therefore, are more concerned with data distributions.

5 RBF NETWORKS FOR GRADE INTERPOLATION

Two main principles – hypotheses have been accepted during the development and application of RBF networks to grade interpolation: it can be approached as a hypersurface reconstruction problem in the spatial co-ordinates input vector space, and grades are the numerical representation of a localised phenomenon (ore deposit) - grades themselves present localised behaviour.

It is generally accepted that the input space characteristics as well as its components play a very important role in the performance of neural networks. The input dimensionality controls to a great extent the overall complexity of the neural network topology as well as the amount of training data required to bring the network performance to acceptable levels. Therefore it is very important to select the inputs from the available data in a way that will help reduce the complexity of the network and at the same time provide the right information for the network to be trained on.

5.1 Case Studies in 2D Co-ordinate Space

The input space defines the way of approaching the required task, in this case grade interpolation. Using the sample co-ordinates, for example, in two dimensions (easting and northing) as inputs to a network with the output being the grade of the sample means that grade is treated as a surface in the 2D co-ordinate space. This approach seems to be the most popular among researchers dealing with this problem and gives very good results in case studies where the samples are given on a 2D plane instead of a full 3-dimensional space.

A single RBF network architecture with two inputs (the sample's co-ordinates) and one output (the sample's grade) is sufficient to model the grade of a 2D grade distribution. The number of RBFs in the network's hidden layer varies with the complexity of the required mapping and with the number of available samples. The number of RBFs, their location in the 2D input space, and their receptive field are the main training parameters.

Results from two of the numerous 2D case studies completed with this simple RBF network architecture are given in the following figures. The simple structure of the network allows the visualisation of the RBF locations in the same space with the samples or the estimated values giving a clear understanding of how the network works. The estimated values have a number of reliability measures assigned to them.

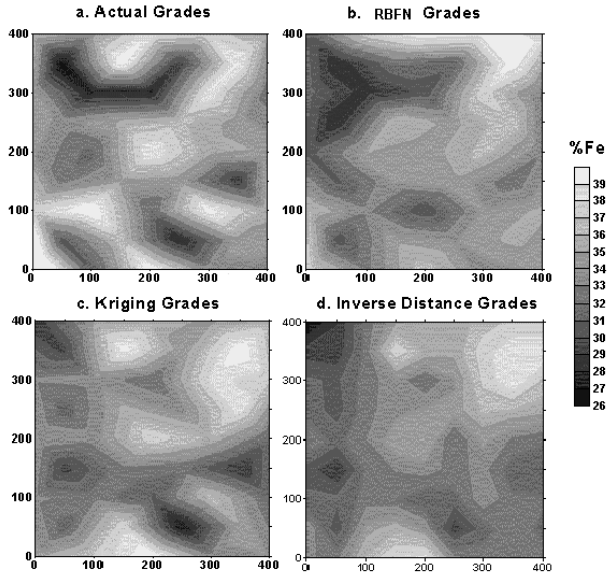

Figure 3: Actual and estimated Fe grades using RBF network, kriging, and inverse distance methods.

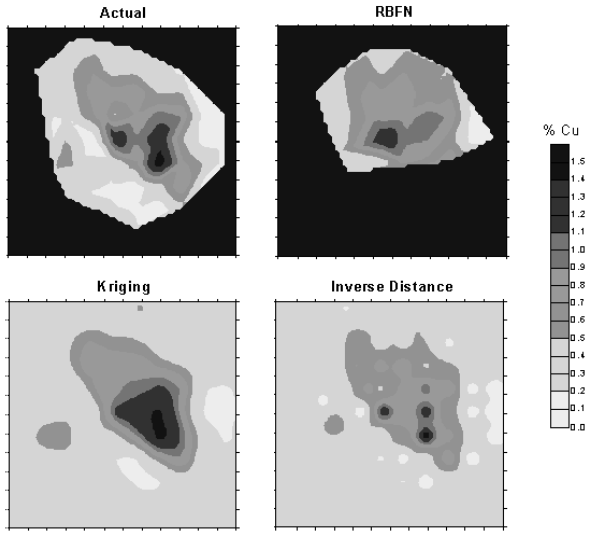

Figure 4: Actual and estimated Cu grades using RBF network, kriging, and inverse distance methods.

Comparison with the inverse distance method and kriging is made on the basis of samples hidden completely from the RBF network training process and excluded from the input data available to these methods. The number of excluded samples varied upon the total number of samples available.

5.2 Case Studies in 3D Co-ordinate Space

Estimation in 3D co-ordinate space is more complex and presents problems when a single-network approach is considered. The complexity of the mapping is, in most cases, beyond the limits of a single RBF network. There are cases

when a single network is sufficient, but in most cases the mapping needs to be broken down into smaller and easier tasks, leading to a modular architecture consisting of several RBF networks. Such an architecture has been developed by the author (Kapageridis 1999) and tested in several case studies from real deposits. Modular architectures have been developed and tested by others with success (Burnett 1995).

The results presented in the graphs below were obtained using the GEMNet II architecture. GEMNet II is a modular neural network system designed by the author to receive drillhole information from an orebody and perform grade estimation on a block model basis.

GEMNet II consists of three modules of Radial Basis Function (RBF) networks. The first module includes six RBF networks each one trying to model the grade's spatial variability in a certain direction in space. second module consists of a single RBF network with four inputs and one output. It is trained using the x, y, z co-ordinates and sample length as inputs and the sample grade as output. In essence, this module 'learns' the spatial distribution of grades. The third module also consists of a single RBF network. This network receives the outputs/grade estimates from the first and second module's networks and performs a final weighting to produce a single estimate.

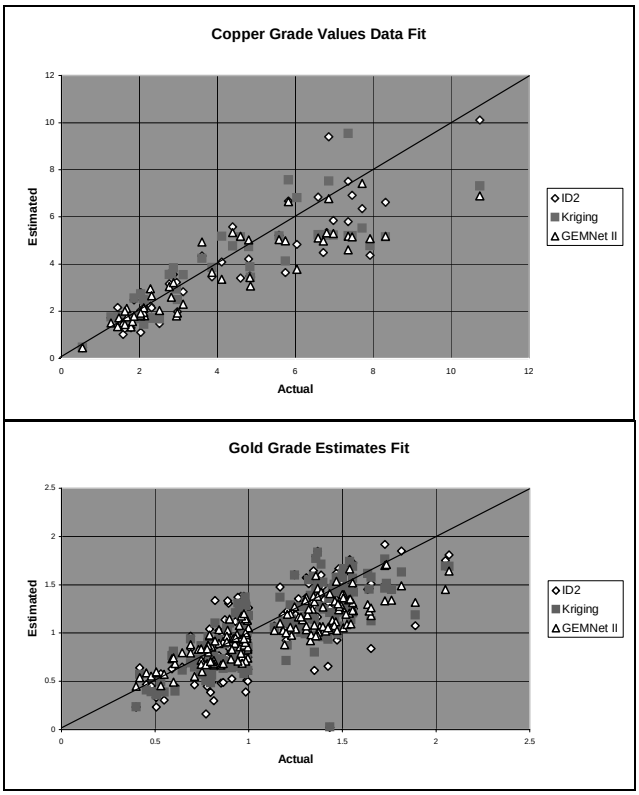

Figure 5: Scatter diagrams of estimated vs. actual copper and gold grades using inverse distance (ID2), kriging, and GEMNet II.

6 CONCLUSIONS

RBF networks, as most of the ANN structures, have certain properties that establish them as a natural choice for grade estimation. However, RBF networks also have a number of additional useful properties that give them an advantage over other ANN architectures for this specific problem.

The first of these properties is that RBF networks construct local approximations to input-output mappings. It is well known that a mineral deposit is a localised phenomenon. Modelling of a deposit's grade in 3D space using drillhole data can be considered to be a problem of hypersurface reconstruction in 3D space, with this hypersurface consisting of a number of zones that need to be locally approximated. Deposits commonly present a localised behaviour; i.e. points within one area of a deposit close to each other tend to have similar grades. Clearly, this area very rarely extends to the entire deposit and, therefore, the approach of fitting RBFs in cleverly chosen locations can be advantageous. These locations are found by clustering of the drillhole data in order to identify these areas of similar grade behaviour.

RBF networks provide an approach to dealing with ill-posed problems due the properties that they inherit from regularisation theory. Grade estimation is an ill-posed problem, even though the underlying phenomenon – the orebody deposition – is well-posed. As was shown, reconstructing a deposit's grade as a hypersurface in the space derived from the drillhole data information, is an ill-posed problem, hence RBF networks should be the choice of ANN for this task.

RBF networks also allow the calculation of reliability measures, such as the *extrapolation measure* and *confidence limit.* Due to the localised nature of approximation performed by RBF networks, it is possible to measure the *local data density* for a given point x in the input space as an index of extrapolation (Leonard *et al.* 1992b). Confidence limits for the model prediction can also be calculated from the local confidence intervals developed for each RBF unit using a weighted average of the latter. These reliability measures were first introduced by Leonard *et al.* (1992a, 1992b) incorporated in a new ANN architecture that computes its own reliability, called the *Validity Index network* (VI). Leonard *et al.* used a two-stage approach based on data densities derived using *Parzen windows* (Parzen 1962), and an interpolation formula used for determining the densities at arbitrary test points. These measures are now standard to most of the commercial neural network simulators that provide RBF network development options.

Finally, another advantage of RBF networks over other ANN architectures that is derived from their theoretical properties, is their speed of development. In the case of low input dimensionality, RBF networks' learning is expected to be a lot faster than in any other ANN architecture used for the same problem. The author approached grade estimation using an input space of maximum four dimensions (Easting, Northing, Elevation, and sample Length), a number low enough for the networks to be very fast to develop.

REFERENCES

Bishop, C.M., Neural Networks for Pattern Recognition. Clarendon Press, Oxford, 1995. 60

Broomhead, D.S., and Lowe, D., Multivariable Functional Interpolation and Adaptive Networks. Complex Systems, Vol. 2, pp 321-355, 1988.

Burnett, C.C.H., Application of Neural Networks to Mineral Reserve Estimation. Ph.D. Thesis, Department of Mineral Resources Engineering, University of Nottingham, 1995.

Chen, S., Nonlinear Time Series Modelling and Prediction Using Gaussian RBF networks with Enhanced Clustering and RLS Learning. Electronic Letters, Vol. 31, No. 2, pp 117-118, 1995.

Chinunrueng, C., and Sequin, C.H., Optimal Adaptive k-means Algorithm with Dynamic Adjustment of Learning Rate. IEEE Trans. On Neural Networks, Vol. 6, pp 157-169. 1994.

Cybenko, G., Approximation by Superpositions of a Sigmoidal Function. Mathematics of Control, Signals, and Systems, Vol. 2, pp 303-314, 1989.

Duchon, J., Spline Minimising Rotation-Invariant Semi-norms in Sobolev Spaces. In: Schempp W., and Zeller, K., (eds), Constructive Theory of Functions of Several Variables, Lecture Notes in Mathematics, pp 85-100, 1977.

Duda, R.O., and Hart, P.E., Pattern Classification and Scene Analysis. Wiley, New York, 1973.

Golub, G.H., and Van Loan, C.G., Matrix Computations, 3rd Edition. Johns Hopkins University Press, Baltimore, 1996.

Haykin, .S., Neural Networks – A Comprehensive Foundation. Prentice Hall, New Jersey, 1999.

Kapageridis, I.K., Application of Artificial Neural Network Systems to Grade Estimation From Exploration Data. Ph.D. Thesis, School of Chemical, Environmental and Mining Engineering, University of Nottingham, 1999.

Kirsch, A., An Introduction to the Mathematical Theory of Inverse Problems. Springer-Verlag, New York, 1996.

Leonard, J.A., Kramer, M.A., and Ungar, L.H., A Neural Network Architecture that Computes Its Own Reliability. Computers Chem. Engineering, Vol. 16, No. 9, pp 819-835, 1992.

Leonard, J.A., Kramer, M.A., and Ungar, L.H., Using Radial Basis Functions to Approximated a Function and Its Error Bounds. IEEE Transactions on Neural Networks, Vol. 3, No. 4, pp 624-627, 1992.

Meinguet, J., Multivariate Interpolation at Arbitrary Points Made Simple. Journal of Applied Mathematics and Physics (ZAMP), 30, pp 292-304, 1979.

Micchelli, C.A., Interpolation of Scattered Data: Distance Matrices and Conditionally Positive Definite Functions. Constructive Approximation, Vol. 2, pp 11-22, 1986.

Moody, J., and Darken, C.J., Fast Learning in Networks of Locally-Tuned Processing Units. Neural Computation, Vol. 1, pp 281-294, 1989.

Morozov, V.A., Regularisation Methods for Ill-Posed Problems. CRC Press, Boca Raton, FL, 1993.

Nadaraya, E.A., On Estimating Regression. Theory of Probability and its Applications, Vol. 9, pp 141-142, 1964.

Park, J., and Sandberg, I.W., Approximation and Radial Basis Function Networks. Neural Computation, Vol. 5, pp 305-316, 1993.

Parzen, E., On Estimation of A Probability Density Function and Mode. Ann. Math. Statist., Vol. 33, pp 1065-1076, 1962.

Poggio, T., and Girosi, F., Regularisation Algorithms for Learning that Are Equivalent to Multilayer Networks. Science, Vol. 247, pp 978-982, 1990.

Powel, M.D., Approximation Theory and Methods. Cambridge University Press, Cambridge, 1981.

Powel, M.D., The Theory of Radial Basis Function Approximation in 1990. In Light, W., (ed.), Advances in Numerical Analysis Vol. II: Wavelets, Subdivision Algorithms, and Radial Basis Functions, pp 105-210, Oxford Science Publications, Oxford, 1992.

Ripley, B.D., Statistical Ideas for Selecting Network Architectures. In: Kappen, B., and Gielen, S., (eds), Neural Networks: Artificial Intelligence and Industrial Applications, Springer, London, 1995.

Singh, S.P., (ed.), Approximation Theory, Spline Functions and Applications. Kluwer, Dordrecht, The Netherlands, 1992.

Tikhonov, A.N., On Solving Incorrectly Posed Problems and Method of Regularisation. Doklady Akademii Nauk USSR, Vol. 151, pp 501-504, 1963.

Tikhonov, A.N., and Arsenin, V.Y., Solutions to Ill-Posed Problems. W.H. Winston, Washington, DC, 1977.

Watson, G.S., Smooth Regression Analysis. Sankya: The Indian Journal of Statistics, Series A, Vol. 26, pp 359-372, 1964.

Yee, P.V., Regularised Radial Basis Function Networks: Theory and Applications to Probability Estimation, Classification, and Time Series Prediction. Ph.D. Thesis, McMaster University, Hamilton, Ontario, 1998.