Radial Basis Function Networks – Revisited

David Lowe FIMA, Aston University

1 History and context

The radial basis function (RBF) network is an adaptive network model introduced by Broomhead and Lowe [1] which was motivated by the mathematics of approximating arbitrary continuous functions. This essay presents a personal recollection and reflection of key ideas in the introduction, motivation and evolution of the RBF network. The original article was slow to be taken up by the community, but has now passed over 3,000 citations. Its popularity probably stems from its simplicity combined with fundamental insights it offers in diverse and difficult problem domains which has allowed it to transcend its original motivation and be applied widely and successfully. For detailed reviews with extensive references see [2] and [3].

In 1986 Dave Broomhead and I were working at the Royal Signals and Radar Establishment in Malvern, UK. He was in the Signal Processing and I was part of the Speech Research Unit. It was at the time an intense environment of stimulating intellectual activity on curiosity- as well as project-driven research, and we were surrounded by many people across the speech, signal and pattern processing domains who have since been responsible for world-leading major advances in these fields. It was an exciting place to work rivalling the best university departments. Although we were in different groups, we had known each other since our time in the Physics Department at Warwick University, and we were unofficially working together on our mutual desire to understand the fundamental science behind the resurgent interest in artificial neural networks. In fact, we were actually working on another pet topic of Dave's – non-linear dynamics. Specifically, exploring a model of interactions between more conventional neuron abstractions of coupled processing nodes using a master-slave decomposition, and Haken's synergetics framework. However, after a typically serendipitous meeting, we were about to drop that direction of work, always intending to revisit it at a later time (unfortunately, we never did).

At that time, the artificial neural network field was struggling with issues of inefficient implementations, and the optimisation (learning, training) and generalisation of these interconnected non-linear pattern processing machines. It was the time of the popularisation of *backpropagation* – the recursive use of the chain rule to evaluate the local function gradient coupled with a simple steepest descent algorithm for reducing an error function by iteratively modulating the many parameters (weights) in these multilayer perceptrons. We were seeking an alternative viewpoint to the empiricism dominating the field.

It was around this time that Dave had returned from one of his overseas research visits excited about a talk he had attended on function interpolation over infinite regular lattices. The talk that inspired Dave was by M.J.D. Powell of the Cambridge Numerical Analysis Group. Regular readers of *Mathematics Today* will recall Mike Powell as a recipient of the Catherine Richards Prize for best article in *Mathematics Today* in 2007 [4]. To complete the cycle of coincidence, it was Mike Powell visiting the RSRE Speech Research Unit in the mid 1980s who made us aware of more appropriate non-linear optimisation approaches for optimising the multilayer neural models, including the conjugate gradient and quasi-Newton algorithms, which we developed and adapted, and that we still use today, and are distributed freely as part of the Aston Netlab software [5]. It is with sadness we note that Professor Powell also recently passed away on 19 April 2015.

Dave's serendipitous attendance at Mike Powell's talk on interpolating functions over lattices, gave us the insight on neural networks that we had been seeking. The formal summary of this thesis was publicly published in *Complex Systems* in 1988 [1] (also released as RSRE Memorandum no. 4148).

2 The radial basis function: strict interpolation and function approximation

The primary motivation was the ongoing fundamental mathematical work into the theory of approximating continuous functions by interpolating across known lattice points, principally driven by Powell [6] and Micchelli [7]. In the case of scattered data interpolation where the lattice points are replaced by distinct observed data points, the formal mathematical question addressed by this activity was:

Problem: *Given a set of m distinct vectors (data points), $\{\boldsymbol{x}_i; i = 1, 2, \ldots, m\}$ in $\mathbb{R}^n$ and m real numbers $\{f_i; i = 1, 2, \ldots, m\}$, find a function $s : \mathbb{R}^n \to \mathbb{R}$ which satisfies the interpolation conditions: $s(\boldsymbol{x}_i) = f_i$, $i = 1, 2 \ldots m$.*

Note that in this problem the function $s(\boldsymbol{x})$ is constrained to *exactly* fit the given data points. This is known as strict function interpolation. The method of solution to this problem taken by Powell et al. using RBFs was to construct a weighted linear combination of non-linear *basis functions* $\phi_i(\boldsymbol{x})$ which were functions of $\boldsymbol{x}$ but centred on the set of specific discrete points, $\boldsymbol{x}_i$, and thus could be expressed as $\phi(\|\boldsymbol{x} - \boldsymbol{x}_i\|)$. The vectors $\boldsymbol{x}_i$ were considered to be the *centres* of the basis functions. In terms of these basis functions, a class of interpolating functions was constructed of the following form:

$$s(\boldsymbol{x}) = \sum_{j=1}^{m} \lambda_j \phi(\|\boldsymbol{x} - \boldsymbol{x}_j\|), \quad \boldsymbol{x} \in \mathbb{R}^n.$$

The unknown parameters λ_j need to be determined by optimising the model to the data points. Intuitively, around each data point we place a basis function overlapping with all other basis functions, weighted by an adaptable parameter to recreate the

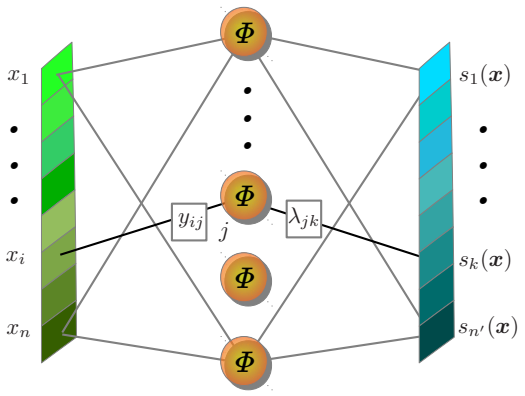

Figure 1: A pictorial representation of the RBF network revealing its role as a feed-forward adaptive network structure.

unknown function assumed to have generated the data points. In other words, a flexible fitting surface has been generated using a finite set of random functions, tunable to a given finite set of data points.

The advantage of this approach was that the solution of the above (generally non-linear!) interpolation problem is specified using simple linear optimisation. We can see this by inserting the above defined radial basis interpolation function into the interpolation conditions (i.e. the requirements that $\boldsymbol{s}(\boldsymbol{x}_i) \equiv f_i$), to give a set of linear (in the unknown parameters λ_j) equations:

$$\begin{aligned} s(\boldsymbol{x}_1) &= \sum_{j=1}^{m} \lambda_j \phi(\|\boldsymbol{x_1} - \boldsymbol{x}_j\|) = f_1 \\ s(\boldsymbol{x}_2) &= \sum_{j=1}^{m} \lambda_j \phi(\|\boldsymbol{x_2} - \boldsymbol{x}_j\|) = f_2 \\ &\vdots \\ s(\boldsymbol{x}_m) &= \sum_{j=1}^{m} \lambda_j \phi(\|\boldsymbol{x_m} - \boldsymbol{x}_j\|) = f_m \end{aligned}$$

or, in matrix form: $\boldsymbol{A\lambda} = \boldsymbol{f}$, where $\boldsymbol{A}$ is a square matrix with elements $A_{ij} = \phi(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|), i, j = 1, 2, \ldots, m$, $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$ and $\boldsymbol{f} = (f_1, \ldots, f_m)$. For a very broad class of basis functions and very general conditions, the interpolating matrix $\boldsymbol{A}$ is non-singular and hence its inverse exists. Thus, the equation can be simply solved to determine the unknown mixing parameters as $\boldsymbol{\lambda} = \boldsymbol{A}^{-1}\boldsymbol{f}$.

This framework easily extends to functions which map into more than one dimension where the desired interpolated points are in n' dimensions instead, i.e. $\boldsymbol{f}_i \in \mathbb{R}^{n'}$, so that $s : \mathbb{R}^n \to \mathbb{R}^{n'}$. This now means that we need a different weighting parameter for each one of the n' dimensions as well. The only difference to the above is that the output values are now vectors instead of scalars, and there is now a vector of parameters to fit each kth dimension as

$$\begin{aligned} \sum_{j=1}^{m} \lambda_{jk} \phi(\|\boldsymbol{x_1} - \boldsymbol{x}_j\|) &= (\boldsymbol{f}_1)_k \\ \sum_{j=1}^{m} \lambda_{jk} \phi(\|\boldsymbol{x_2} - \boldsymbol{x}_j\|) &= (\boldsymbol{f}_2)_k \\ &\vdots \\ \sum_{j=1}^{m} \lambda_{jk} \phi(\|\boldsymbol{x_m} - \boldsymbol{x}_j\|) &= (\boldsymbol{f}_m)_k. \end{aligned}$$

Again, as a matrix equation this can be written as $\boldsymbol{A\Lambda} = \boldsymbol{F}$, where $\boldsymbol{A}$ is defined as before, and now $\boldsymbol{\Lambda}$ is a square matrix of fitting parameters (instead of a vector) $\boldsymbol{\Lambda}_{jk} = \lambda_{jk}$, and $\boldsymbol{F} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_m)$ is the set of m vectors to be interpolated. Again, the unknown parameters can be obtained by matrix inverse techniques: $\boldsymbol{\Lambda} = \boldsymbol{A}^{-1}\boldsymbol{F}$.

However, even though the set of *training* patterns $\{(\boldsymbol{x}_i, \boldsymbol{f}_i), i = 1, \ldots, m\}$ determines the fitting parameters directly, this approach has several disadvantages, conceptually and computationally. Note that its complexity increases linearly with the volume of data since there is one RBF centre used for each input data point, and so for larger networks the computational cost increases non-linearly (due to the increased cost of computing an inverse matrix). However the main disadvantage arises when this approach is used to analyse real data, for which we need the machine learning version of the RBF network.

3 The radial basis function network model: approximate interpolation

A significant conceptual downside of the approach is that the requirement to exactly fit a finite set of data points, also means that any noise fluctuations in the observations would distort the fitting surface to unreasonably pass through all the data points, including all the random noise fluctuations. Therefore, to accommodate noise and uncertainty in observations, the strict interpolation conditions needed to be relaxed, since otherwise the fitting function will exactly fit all the noise fluctuations rather than approximate the data generator of interest.

So the suggestion was to distinguish the centres from the m observed data points and thus allow a different number, n_0 of centres than data points ($\boldsymbol{y}_j, j = 1, \ldots, n_0$, where typically $n_0 < m$). These n_0 centres could now be selected or optimised at convenient positions in data space to represent the data distribution. This has the effect of regularising the fitting function producing a smoother interpolation surface in the data space. Now the interpolating problem becomes over specified, the interpolating matrix $\boldsymbol{A}$ above is no longer square and the previous unique exact matrix inverse solution becomes an approximate linear optimisation problem.

An additional useful modification was to introduce a set of constant offsets to compensate for the *bias* in each output dimension, λ_{0k}, into the framework. These compensate for the difference between the average value over the data set of the RBF network outputs and the corresponding average value of the targets.

This led to the machine learning version of the RBF network in the form

$$\boldsymbol{s}_k(\boldsymbol{x}) = \sum_{j=0}^{n_0} \lambda_{jk} \phi(\|\boldsymbol{x} - \boldsymbol{y}_j\|), \quad k = 1, \ldots, n', \qquad (1)$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{y}_j$ are the selected basis function centres (fixed, and now generically distinct from the data points $\boldsymbol{x}$) and $\phi(\|\boldsymbol{x} - \boldsymbol{y}_0\|) \equiv 1$ and is visually represented in Figure 1.

In this form the RBF approach maps over to an adaptive network model in which the centres form the (fixed) parameters of the first layer of the network, the weights and biases λ_{jk} form the parameters of the final layer weights, and the basis function $\phi(\ldots)$ represents the non-linearity on each *hidden* node. Instead of a scalar product between the input data pattern and the first-layer weights as would typically be used in a multilayer perceptron neural network, the RBF network used a distance function $\|\boldsymbol{x} - \boldsymbol{y}\|$ which has often been assumed to be Euclidean, but this is not essential and indeed it need not even be a positive-definite metric. Similarly, the basis function has often been assumed to be local and Gaussian. However, this assumption is incorrect and indeed can be a bad choice due to its very poor numerical convergence properties in dense systems. In the original paper we demonstrated analytic solutions to classically *hard* problems using non-local basis functions, as well as the Gaussian non-linearity. Using non-local (and non positive definite) basis functions still allows the network response as a whole to locally interpolate the data space and with improved numerical convergence.

One of the advantages of the RBF network is that the first-layer weights $\{\boldsymbol{y}_j, j = 1, \ldots, n_0\}$ may often be determined or specified by a judicious use of prior knowledge, or estimated by simple techniques permitting the linear optimisation of the model.

Figure 2 depicts a simple example of the interpolation properties of the RBF network which illustrates a set of 20 data points sampled from a sine wave and corrupted by additive noise. Using just these data samples, the output of the optimised RBF *for different numbers of randomly chosen spline basis functions* is shown for different network complexities. For 20 centres the RBF fits exactly all the data points, which includes all the random noise fluctuations as well. For just three centres selected, the fit is almost a straight line, and for seven centres the RBF is closely approximating the original generating function. The complexity of the network is one way used to regularise the RBF network solution.

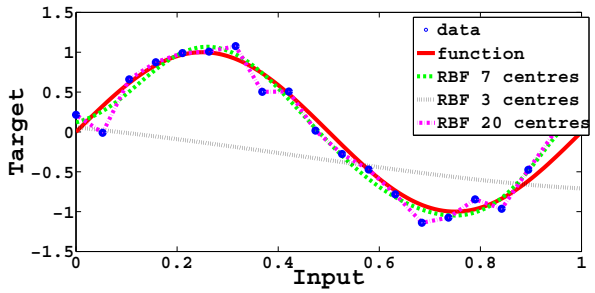

Figure 2: Simple illustration of interpolating a function (a sine wave) using a RBF network based on 20 noisy observations generated from the sine wave. The figure shows under- and over-fitting as the complexity of the model is varied.

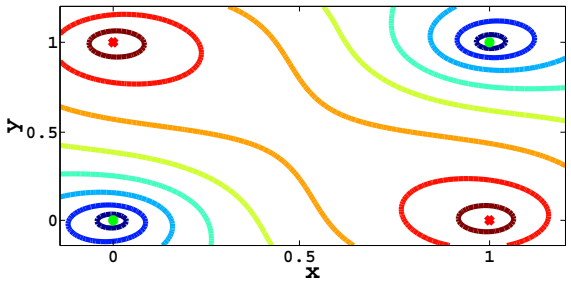

Figure 3: The RBF network interpolation function on the classical XOR problem. 'X' (✱) denotes even parity class and are contained within the 0.1 blue-spectrum contour lines, and 'O' (●) represents the odd parity class contained within the 0.9 red-spectrum contour boundary lines. This is not linearly separable, but is separable by the RBF as shown by the contour lines.

In our original paper we analytically derived the interpolating problem appropriate for the XOR problem. The XOR problem is a classic example of a non-linearly separable problem. However, Figure 3 depicts a RBF interpolating surface which correctly separates out the even and odd parity points, showing both its capacity for clustering as well as its mode of solution of using interpolating surfaces constructed from random non-linear functions. In this figure the interpolating function exactly passes through the four data points with a value of 0 on the surface if the input points are even parity, and a value of 1 if the two input values are of odd parity. The interpolating surface represents a continuous generalisation of the binary XOR problem.

So, as an adaptive network model, the RBF network provided insight at several levels. *Learning* by neural networks became *curve fitting* to a high-dimensional surface generating the observed data. Thus, the awkward issue of *generalisation* in such learning machines was then posed as *interpolation* between the known data points along the surface defined by the RBF network. Moreover, the tricky problem of multiple local minima in the gradient descent algorithms of training multilayer perceptron models leading to unsatisfactory solutions, was instantly circumvented in the class of RBF networks since, although it could universally approximate arbitrary non-linear functions, its optimisation algorithm had the known convergence properties of linear least squares optimisation.

The amazing thing is that this simple RBF approach to function interpolation is computation universal [8] (and in fact *any* non-pathological function on a bounded domain may be interpolated using a *finite* number of such basis functions), has known mathematical convergence properties, and is amenable to efficient computational techniques with known globally optimal properties.

4 Conclusion

This brief article has provided a personal view on the original motivations and introduction of the RBF network, with some historical perspective and context. Its longevity is not due to its capability, but more to the fundamental insights and interpretations that the overall framework provides. This has allowed it to evolve beyond its original motivations from deterministic function fitting, to include statistical perspectives, machine learning insights, topographic visualisation and, more recently, as a cascaded model for deep layer big data classification [9]. The RBF network still has insights to deliver.

Would Dave Broomhead have been surprised? I suspect not.

References

1 Broomhead, D.S. and Lowe, D. (1988) Multi-variable functional interpolation and adaptive networks, *Complex Syst.*, vol. 2, pp. 321–355.

2 Buhmann, M.D. (2003) *Radial Basis Functions: Theory and Implementations*, Cambridge University Press.

3 Wu, Y., Wang, H., Zhang, B. and Du, K.-L. (2012) Using radial basis function networks for function approximation and classification, *ISRN Appl. Math.*, vol. 2012, no. 324194, 34 pages.

4 Powell, M.J.D. (2007) A view of algorithms for optimization without derivatives, *Math. Today*, vol. 43, no. 5, pp. 170–174.

5 `www.aston.ac.uk/eas/research/groups/ncrg/resources/netlab/overview-and-examples/`

6 Powell, M.J.D. (1990) The theory of radial basis function approximation in 1990, in *Advances in Numerical Analysis, vol. II, Wavelets, Subdivision Algorithms, and Radial Basis Functions*, ed. Will Light, Oxford Science Publications, Clarendon Press, pp. 105–210.

7 Micchelli, C.A. (1986) Interpolation of scattered data: distance matrices and conditionally positive definite functions, *Constructive Approximation*, vol. 2, pp. 11–22.

8 Park, J. and Sandberg, I.W. (1991) Universal approximation using radial basis function networks, *Neural Comput.*, vol. 3, pp. 246–257.

9 Rice, I. and Lowe, D. (2014) Deep layer radial basis function networks for classification, *10th International Conference on Mathematics in Signal Processing*, Institute of Mathematics and its Applications.