Differential Evolution Algorithm as a Tool of Training Radial Basis Function Networks

Junhong Liu **Jouni Lampinen**

Department of Information Technology
Lappeenranta University of Technology
P.O.Box 20, FIN–53851 Lappeenranta
Email: liu@lut.fi, jlampine@lut.fi

Abstract. The Differential Evolution (DE) algorithm is a floating-point encoded evolutionary algorithm for global optimization. It has been demonstrated to be an efficient, effective, and robust optimization method especially for problems containing continuous variables. This paper concerns applying DE to training the radial basis function (RBF) networks. It is demonstrated by training networks to approximate three nonlinear functions. The Euclidean distance from the desired outputs to the actual network outputs is applied as the objective function to be minimized. The process converges effectively. The results show that DE is a potential way to train Gaussian RBF networks.

Keywords: Radial basis functions, Differential evolution, Neural networks, Evolutionary algorithms

1 INTRODUCTION

Radial Basis Functions (RBFs) emerged as a variant of artificial neural networks (ANNs) in the late 80's. RBFs are embedded in a three layer NN, where each hidden unit implements a radial activation function. The output units implement a weighted sum of hidden unit outputs. Approximation capabilities of RBFs have been studied in [1, 2]. Due to their nonlinear approximation properties, RBF networks are able to model complex mappings, but perceptron NNs can only model these complex mappings by means of multiple intermediary layers [3, 4]. RBFs have been used to build the class of nonlinear models, i.e. RBF models, for multivariate approximation, partially because the RBF models have the properties of localization, boundedness, stability, good interpolation, smoothness.

The performance of a trained RBF network depends on the number and locations of the RBFs, their shapes, and the method used for learning the input-output mapping. Finding the RBF weights is called network training. With an existing set of input-output pairs, called the training set, the network parameters are optimized in order to fit the network outputs to the given inputs. The fit is evaluated by means of a cost function. After training, the RBF network can be used to respond to data whose underlying statistics is similar to that of

the training set. A variety of approaches for training RBF networks have been developed, which can be divided into three categories: (*i*) learning the centres [5–7] and widths in the hidden layer [5]; (*ii*) learning the connection weights from the hidden layer to the output layer [5–10]; (*iii*) learning the network structure [5–11]. RBF networks have been successfully applied to a large diversity of applications including interpolation [12, 13], signature recognition [9].

ANNs are widely recognized for their ability to approximate complicated non-linear relationships and to estimate underlying trends, even when substantial noise is present in the data. But it is often difficult to design appropriate NN models since the basic principles governing the processing of information in NNs are not well understood. Because the complex interactions among network units usually make conventional design techniques inapplicable, more efficient methods are required for the development of neural processing systems [11]. Evolutionary techniques fit this purpose. Evolutionary algorithms (EAs) have been successfully applied to finding the global optima of various multidimensional functions where local optima in the space of possible solutions are common and EAs are able to handle the optimization of parameters for which no gradient information exits or are needed. The Differential Evolution (DE) algorithm introduced by Storn and Price [14], a floating-point encoded EA for global optimization, is used in the paper to find the set of parameters of RBFs to provide the best possible function approximation.

The paper is structured as follows: the formulation of optimization problem is explained briefly in Section 2, the optimization system of RBF networks using Differential Evolution is described in Section 3, and experimental results are shown in Section 4. Conclusion is given in Section 5.

2 OPTIMIZATION PROBLEM FORMULATION

2.1 Radial Basis Functions Network

RBFs have their origin in the solution of the multivariate interpolation problem [12]. They can approximate an arbitrary function $g(\mathbf{v})$: $\Re^d \rightarrow \Re$ by mapping using a RBF network with a single hidden layer of p units:

$$\widehat{g}(\mathbf{v}, \mathbf{x}) = \omega_0 + \sum_{j=1}^{p} \omega_j r_j(\mathbf{v}, \boldsymbol{\sigma}_j, \mathbf{c}_j) = \omega_0 + \sum_{j=1}^{p} \omega_j \phi_j(\boldsymbol{\sigma}_j, \|\mathbf{v} - \mathbf{c}_j\|), \qquad (1)$$

where $\mathbf{v} \in \Re^d$; $\mathbf{x}$ is a vector including all variable factors w_0, w_j, $\boldsymbol{\sigma}_j$, and $\mathbf{c}_j$; w_0 is the bias; $\mathbf{w} = (w_1, w_2, ..., w_p)^{\mathrm{T}}$ are the weights coefficients; p denotes the number of the basis functions; $\mathbf{c}_j$, $j = 1, 2, ..., p$, are the centres; $\boldsymbol{\sigma}_j$, $j = 1, 2, ..., p$, are the widths, which are called scaling factors for the radii $\|\mathbf{v} - \mathbf{c}_j\|$; $r_j(\cdot)$ represents the activation function (radial basis function) from $\Re^d$ to $\Re$. In order to simplify the notation we use coordinate axes-aligned Gaussian RBF functions. When a $2D$ Gaussian RBF is centred at the centroids $\mathbf{c}_j$, it follows from (1) that

$$\widehat{g}(\mathbf{v}, \mathbf{x}) = \omega_0 + \sum_{j=1}^{p} \omega_j e^{-\frac{\|\mathbf{v}-\mathbf{c}_j\|^2}{2\sigma_j^2}} = \omega_0 + \sum_{j=1}^{p} \omega_j e^{-\frac{(v_1-c_{j1})^2}{2\sigma_{j1}^2}} e^{-\frac{(v_2-c_{j2})^2}{2\sigma_{j2}^2}}, \qquad (2)$$

where $\mathbf{x}$ is the vector of all variable factors, can be written as

$$\begin{aligned}\mathbf{x}^{\mathrm{T}} &= (w_0, \mathbf{w}^{\mathrm{T}}, \boldsymbol{\sigma}_1{}^{\mathrm{T}}, \mathbf{c}_1{}^{\mathrm{T}}, ..., \boldsymbol{\sigma}_j{}^{\mathrm{T}}, \mathbf{c}_j{}^{\mathrm{T}}, ..., \boldsymbol{\sigma}_p{}^{\mathrm{T}}, \mathbf{c}_p{}^{\mathrm{T}})^{\mathrm{T}} \\ &= (w_0, w_1, ..., w_p, \sigma_{11}, \sigma_{12}, c_{11}, c_{12}, ..., \sigma_{j1}, \sigma_{j2}, c_{j1}, c_{j2}, ..., \sigma_{p1}, \sigma_{p2}, c_{p1}, c_{p2})^{\mathrm{T}}.\end{aligned}$$

The network can be trained to approximate an unknown function $g(\mathbf{v})$ by finding the optimal vector $\mathbf{x}$ given a (possibly noisy) training set $V = \{(\mathbf{v}_n, y_n) | n = \{1, 2, ..., N\}, \mathbf{v}_n \in \Re^d\}$.

2.2 Objective Function

Given the number of radial basis functions and a training set, the network parameters are found such that they minimize the Euclidean distance between the desired and actual outputs, i.e. the objective function:

$$f(\mathbf{x}) = \sqrt{\sum_{n=1}^{N} \Big(y_n - \widehat{g}(\mathbf{v}_n, \mathbf{x}) \Big)^2}. \tag{3}$$

2.3 Brief Description of the Differential Evolution Algorithm

The optimization target function is of the form

$$f(\mathbf{x}) : \Re^D \to \Re. \tag{4}$$

The optimization objective is to minimize the value of the target function by finding the optimal values of its parameters, $\mathbf{x}$, a vector composed of D objective function parameters. Usually the parameters of the target function are also subject to the lower and upper boundary constraints $\mathbf{x}^{(\mathrm{L})}$ and $\mathbf{x}^{(\mathrm{U})}$: $x_k^{(\mathrm{L})} \leq x_k \leq x_k^{(\mathrm{U})}, k = 1, ..., D$.

DE is a parallel direct search method that utilizes D-dimensional parameter vectors, $\mathbf{x}_{i,G}, i = 1, 2, ..., N_{pop}$, as a population for the generation G, where N_{pop}, the number of population members, does not change during the optimization process, to minimize the target function.

3 OPTIMIZATION OF RADIAL BASIS FUNCTION NETWORKS BY DIFFERENTIAL EVOLUTION

3.1 Representation of Parameters of the Objective Function

When a single-input and single-output function is computed using a set of 1-dimensional Gaussian RBF functions, the optimization process will minimize the objective function in (3), by finding an optimal parameter vector:

$$\mathbf{x}^{\mathrm{T}} = (\omega_0, \omega_1, ..., \omega_j, ..., \omega_p, \sigma_{11}, c_{11}, ..., \sigma_{j1}, , c_{j1}, ..., \sigma_{p1}, c_{p1},)^{\mathrm{T}}. \tag{5}$$

The dimensionality of the parameter vector, D, is $3p+1$. Especially when $p = 1$ and $\mathbf{x}^{\mathrm{T}} = (0 \quad 1 \quad 0.5 \quad 0.1)^{\mathrm{T}}$, (2) becomes $\widehat{g}(v, \mathbf{x}) = e^{-\frac{(v-0.5)^2}{2\times 0.1^2}}$ as shown in Fig. 1a.

3.2 Control Parameters' Setting

Parameters of the objective function usually correspond to the network architecture. The evolutionary process starts with a population of these parameters randomly generated based on the defined region and the boundaries of the discussed function, i.e. $c_j \in [-0.25 + v^{(\mathrm{L})}, 0.25 + v^{(\mathrm{U})}]$, $\sigma_j \in (0, (v^{(\mathrm{U})} - v^{(\mathrm{L})})/2]$, and $\omega_j \in [-2 \times |g|_{max}, +2 \times |g|_{max}]$, where $v^{(\mathrm{L})}$ and $v^{(\mathrm{U})}$ are the lower and upper limits of v, and $|g|_{max}$ is the maximum of the absolute values of the function $g(v)$. The control parameters' setting affects the performance of DE, and its values were chosen based on discussions in [15], and are given in Table 1.

Table 1. Control parameters' setting

Variable	Value
Strategy	DE/rand/1/bin
Number of individuals: N_{pop}	$p \cdot 3 \cdot 10$
Crossover operator: C_r	0.9
Mutation amplification: F	0.9

4 EXPERIMENTAL RESULTS AND ANALYSIS

The proposed system was applied to three test functions without any disturbance: (*i*) a Hermite polynomial, given by $g_1(v) = 1.1(1 - v - 2v^2)e^{(-\frac{v^2}{2})}, v \in [-4, +4]$, as shown in Fig. 1b; (*ii*) a $1D$ sine wave function, given by $g_2(v) = sin(12v), v \in [0, 1]$, as shown in Fig. 1c; (*iii*) the third function, given by $g_3(v) = sin(20v^2), v \in [0, 1]$, as shown in Fig. 1d. The data set, V, contains N uniformly spaced noiseless points in the defined region. The DE algorithm ran with the control parameters' setting in Table 1. The experiments were repeated 10 times. Key experimental results are illustrated in the following:

1. Fig. 1b–d show optimization examples: the three functions, the approximation results, the composing RBF functions, and the centres of RBFs.
2. Table 2 reveals: (*i*) after 4×10^4 generations, the objective function value, $f(\mathbf{x})$, decreases from 10.0053 to 0.0384 for $g_1(v)$ and from 8.2830 to 0.0969 for $g_2(v)$, and the approximation error is under 1.5×10^{-3} ($g_1(v)$) and 4.5×10^{-3} ($g_2(v)$) while the standard deviations of approximation are 3.3666×10^{-4} and 7.5326×10^{-4} for $g_1(v)$ and $g_2(v)$ individually; (*ii*) after 1×10^5 generations $f(\mathbf{x})$decreases from 6.4621 to 3.9234 for $g_3(v)$ with the approximation error in the range of $[0.0026, 0.8825]$ and the standard deviation 0.2016; (*iii*) the proposed system worked more efficiently in the first two functions than in the last one which has a changeable frequency and turns more sharper.

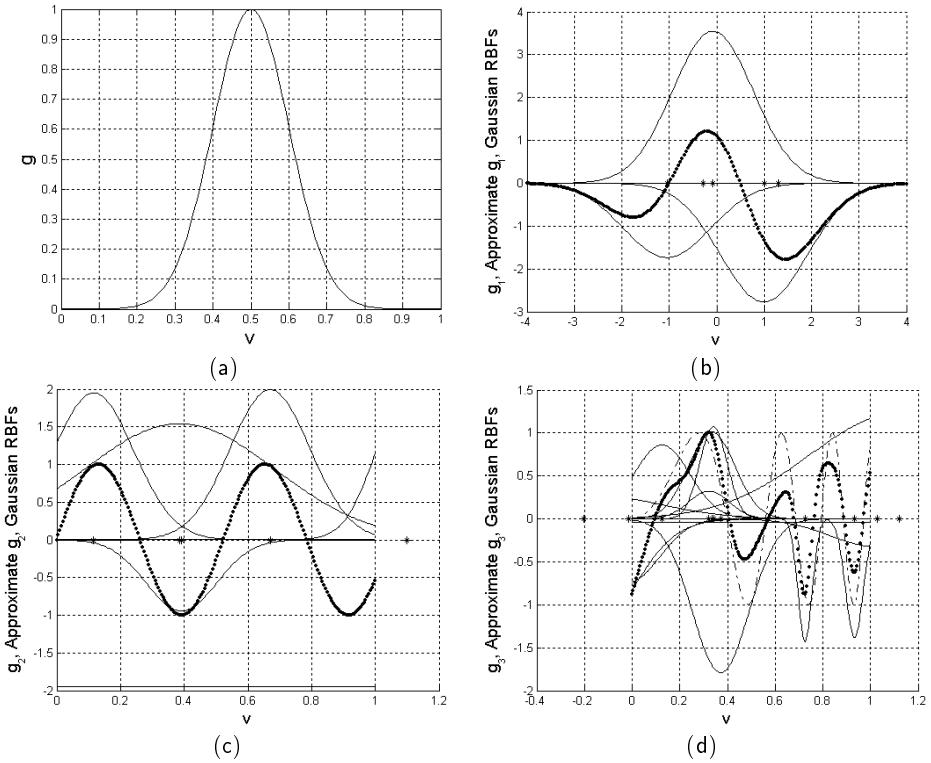

Fig. 1. (a) A $1D$ Gaussian RBF centred at 0.5 with width 0.1 and weight 1; (b) A Hermite polynomial, its approximation and decomposition of a RBF network (5 centroids). (c) A $1D$ sine wave function, its approximation and decomposition (5 centroids) (d) The third function, its approximation and decomposition (12 centroids) (figure legend: "*" — centres; "-" — Gaussian functions and bias; "·" — approximations; "– -" — original functions)

Table 2. Experimental results

Function	N	p	Initial $f(\mathbf{x})$	Final $f(\mathbf{x})$	Generations	$\lvert g_i(v) - \widehat{g}_i(v, \mathbf{x})\rvert$	std
$g_1(v)$	201	5	10.0053	0.0384	4×10^4	$[0, 1.5 \times 10^{-3}]$	3.3666×10^{-4}
$g_2(v)$	201	5	8.2830	0.0969	4×10^4	$[0, 4.5 \times 10^{-3}]$	7.5326×10^{-4}
$g_3(v)$	101	12	6.4621	3.9234	1×10^5	$[0.0026, 0.8825]$	0.2016

Note: "*std*" stands for standard deviation; the approximation error and standard deviation are based on 201 evenly distributed points in the defined region.

5 CONCLUSION

The Differential Evolution algorithm was applied to training Radial Basis Function networks for approximating three nonlinear functions. The choice of the optimal network parameters corresponds to the minimum Euclidean distance between the desired network outputs and actual network outputs. The obtained results suggest that the Differential Evolution algorithm is effective in optimizing the random parameters of Gaussian-type RBF networks.

The future research should include tests with higher data dimensionalities and with less smooth data as well as discover possibilities towards improving the efficiency of the Differential Evolution based training process.

References

[1] T. Poggio and F. Girosi, Networks for approximation and learning, *Proc. IEEE*, MIT, Cambridge, MA, USA, September 1990, **78**(9) (1990) 1481–1497

[2] J. Park and J. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, **3** (1991) 246–257

[3] S. Haykin, *Neural networks: a comprehensive foundation*, Macmillan College Publishing Company, New York (1994)

[4] A. Esposito, M. Marinaro, D. Oricchio, and S. Scarpetta, "Approximation of continuous and discontinuous mappings by a growing neural RBF-based algorithm," *Neural Networks*, **13** (2000) 651–665

[5] A. Leonardis and H. Bischof, "An efficient MDL-based construction of RBF networks," *Neural Networks*, **11** (1998) 963–973

[6] Z. Wang and T. Zhu, "An efficient learning algorithm for improving generalization performance of radial basis function neural networks," *Neural Networks*, **13** (2000) 545–553

[7] Alexandridis, H. Sarimveis, and G. Bafas, "A new algorithm for online structure and parameter adaptation of RBF networks," *Neural Networks*, **16** (2003) 1003–1017

[8] S. A. Billings and G. L. Zheng, "Radial basis function networks configuration using genetic algorithms," *Neural Networks*, **8**(6) (1995) 877–890

[9] Q. Zhu, Y. Cai, and L. Liu, "A global learning algorithm for a RBF network," *Neural Networks*, **12** (1999) 527–540

[10] V. P. Plagianakos and M. N. Vrahatis, Neural network training with constrained integer weights, *Proc. 1999 Congress on Evolutionary Computation*, Washington, DC USA, 6-9 July 1999, **3** (1999) 2007–2013

[11] G. P. J. Schmitz and C. Aldrich, "Combinatorial evolution of regression nodes in feedforward neural networks," *Neural Networks*, **12** (1999) 175–189

[12] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Systems*, **2** (1988) 321–355

[13] M. T. Musavi, W. Ahmed, K. H. Chan, K. B. Faris, and D. M. Hummels, "On the training of radial basis function classifiers," *Neural Networks*, **5** (1992) 595–603

[14] R. Storn and K. Price, "Differential Evolution - a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization* **11**(4) (December 1997) 341–359.

[15] K. Price and R. Storn, "Differential Evolution - a simple evolution strategy for fast optimization," *Dr. Dobb's Journal* **22**(4) (April 1997) 18–24 and 78