7th WSEAS Int. Conf. on MATHEMATICAL METHODS and COMPUTATIONAL TECHNIQUES IN ELECTRICAL ENGINEERING, Sofia, 27-29/10/05 (pp42-50)

Using Rational B-Spline Neural Networks for Curve Approximation

TANG VAN TO and TANAWAT KOSITVIWAT
Department of Computer Science
Faculty of Science and Technology
Assumption University, Bangkok
Thailand

Abstract: Rational B-spline neural network (RBNN) is a neural network can be used for curves and surfaces approximation using rational B-spline model. The approximation is solved by learning process of rational B-spline neural networks from observation data points. A hybrid genetic based algorithm for optimizing knots, control points and weights of RBNN

Keywords: Rational B-spline, Non-Uniform Rational B-spline, Neural Networks, B-spline Neural Networks, Rational B-spline Neural Network, Curve Approxination, Surface Approximation

1. Introduction

B-spline Neural network (BNN) has been applied well to approximate curves and surfaces [1]. In that paper, the authors defined a B-spline Neural Network to model, to approximate, to learn a B-spline curve. It means that given observation data points along a curve, the B-spline Neural Network will approximate the given curve by a B-spline curve, that is finding the B-spline control points for the approximate curve if knot sequence has been given. However, if the curve is too sharp, the fixed knot point approach may not produce a good approximation curve because when the curve change faster more segments should be used. In order to achieve this goal, Vien and To [1] tried to used the genetic approach to obtain the good knot sequence. Thus, B-spline Neural Network with Genetic learning framework proposed in [1] has been a good tool to approximate the complex curves and also surfaces because it does not only finding the good control points and also determine the good knot sequence. They also extended their approach to model tensor product surfaces, that is Non-Uniform B-spline Neural Network (NUBNN) for Tensor Product Surfaces.

In geometric modeling, rational B-spline model also plays a very important role because we have an extra way to control the shape of curves and surfaces, that are the weights [2,4,7,9]. Weight is the level that one control point affects to the local shape of the curve Furthermore, as it has been shown by Piegl [8], a non rational model fail to model exactly conics meanwhile a rational quadratic Bezier curve can do . It is proposed in this paper a model for Rational B-spline Neural Network (RBNN) for learning the process of geometric approximation for curves and surfaces.

The following section briefly introduces B-spline curve, rational B-spline curve and describes the geometric modeling problem. Section 3 describes the topology of rational B-spline neural networks and it application models to approximate geometric objects. Several experiments are given in section 4 to illustrate the achievable results. A short summary of the paper is given in section 5.

2. Fundamentals

Geometric representation of curve and surface by B-spline bases has its roots in approximation theory. The theory of B-spline approximation, primarily, is studied by Schoenberg [2], de Boor [12] and Cox [13]. In following part a very brief of B-spline, rational B-spline, BNN. Then, the rational B-spline approximation problem and related works are given

2.1 B-spline Curves

A B-Spline curve is a piecewise polynomial curve defined by its control points and a knot sequence. The domain is subdivided by knots and basis functions are non-zero on the entire interval. It is defined by:

$$S(t) = \sum_{i=0}^{n} d_i N_{i,n}(t)$$

where $S(t)$ is called a B-spline curve

d_i are the control points of the B-spline curves

$N_{i,n}(t)$ are the i^{th} basic functions of degree n defined over a non decreasing knot sequence $\{ t_i \}$, they are recursively defined as :

$$N_{i,k}(t) = \begin{cases} 1, & if\ t \in [t_i, t_{i+1}) \\ 0, & otherwise \end{cases}$$

$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k} - t_i} N_{i,k-1}(t) + \frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}} N_{i+1,k-1}(t)$$

Knots and a control points only affect some local segments and consequently, a segment is only affected by some knots and neighbor control points.

2.2 Rational B-Spline curves (RBS)

A RBS curve adds a weight w_i to control point d_i and has an equation of

$$S(t) = \frac{\sum_{i=0}^{n} N_{i,n}(t) w_i d_i}{\sum_{j=0}^{n} N_{i,n}(t) w_i}$$

where w_i are the *weights*, that is the affect degree of control points to the shape of the curve. We assume that all the weights are nonnegative number. Obviously, it can be written as:

$$S(t) = \sum_{i=0}^{n} R_{i,n}(t) P_i$$

where

$$R_{i,n}(u) = \frac{N_{i,n}(t) w_i}{\sum_{j=0}^{n} N_{i,n}(t) w_j}$$

$R_{i,n}(t)$ are RBS basis functions. If $w_i = 1$ for all i, then $R_{i,n}(t) = N_{i,n}(t)$ for all i. This shows that the B-spline basis functions are special cases of the rational basis functions. The rational curve in R^d can be considered as a projection of the B-spline curve generated by $\{[w_i d_i; w_i]\}$ in R^{d+1} on the hyperplane of $X_{d+1}=1$ [14].

2.3 B-spline Neural Network

BNN for function approximation was proposed by Brown and Harris [11]. This networks can be considered as a type of feed-forward networks. A typical structure of a B-spline network contains three layers which are a input layer, a hidden layer consists of B-spline basis functions that are defined on the lattice formed by normalising the input space, and the output layer that sums the weighted outputs from the basis functions to produce the network output as shown in Fig. 1.

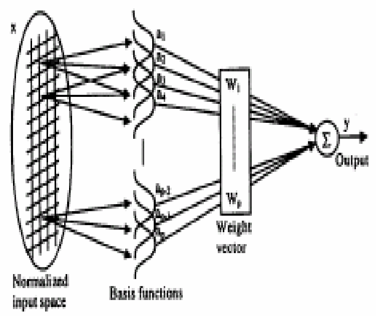

Figure 1: Typical Structure of B-spline Neural Fuzzy Network [11]

2.4 BNN for Curves and Surface Approximation

Vien and To [1] used BNN for approximation curves and surfaces. They had proposed two models fixed knot BNN networks and free knot BNN networks.

In the fixed knot BSN network, the knot sequence has been given, therefore the neural network need to learn weight vector, that really are control points of geometric objects. They used gradient decent algorithm for adjusting control points to minimized the sum square error. To select a good knot sequence for approximate a geometric object is difficult, normally we use a little of segments for the part of curve that has little change.

Rather than either select manually a suitable set of knots or apply directly any fixed knot strategies, i.e., uniform knot allocation, it would efficiently to automatically optimize knot positions for modeling. The earliest numerical algorithms attempting to solve these problem is due to de Boor in [12]. The approach is known as free knot spline approach. Similar works has also been carried out in [3,5,6] .

A hybrid genetic algorithm proposed in [1] to optimize the parameters of BNN networks for geometric modeling problem. The algorithm attempts to separate linear and nonlinear parameters of the BNN for training.

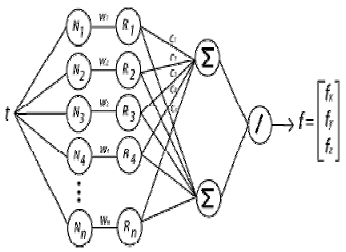

Figure 2: RBSNN Model for curve modeling

Linear weight parameters (control points) are identified by using linear least square estimator method. This approach gives a fast convergence on estimating linear weights. Nonlinear knot parameters are determined by evolution process of genetic algorithms. The process is able of finding global optimal solution to nonlinear problem without using mathematical model restriction.

Hybrid Genetic Algorithms for free knot BSN networks:

Step1: Initialize a knots population with randomly initialized knot chromosomes.
Step2: Evaluate the knot population based on fitness function to determine the best knot chromosome.
Step3: Estimate the linear coefficient weight parameters of BNN networks based on current best knot chromosome by least square estimator method.
Step4: Compute the mean square error of the networks.
S**tep5**: Perform evolution operators on the set of knot chromosomes.
Step6: Repeat step 2, 3, 4 and 5 until converge criterion is met.

3. Rational B-spline Neural Networks Model

3.1: RBNN Architecture

The architecture of RBSNN was suggested in [1] for approximation improvement is given in Fig. 2 This network is the generalization of BNN by exploiting the use of Rational Basis Function in a hidden layer, so the BNN becomes RBNN.

3.2 Fixed Knot RBNN for Curve Approximation

The problem of RBNN for approximation is focused to the determination of the weight of control points. Under a given set of weights, the neural network learns to find control points, a straight method is used the gradient decent algorithm. Weights are obtained by genetic algorithm [10]. The formal algorithm is given as follow:

Step 1 : Initialize weights for control points population with randomly generated weight chromosomes.
Step 2 : For each chromosome, using gradient decent learning method to find the best control points and evaluate the Mean Square Error (MSE)
Step 3: Using Genetic algorithm to produce the new generation of chromosome. Repeat step 2 and 3 until terminate condition is met

3.2.2 Free Knot RBNN for Curve Approximation

For free knot RBNN, we need to determine the control points, weights and the location of knot sequences at a same time. For control points, we used gradient descent method to minimize the MSE, meanwhile for weights and knot sequence are obtained from genetic approach. The algorithm for free knot RBNN is described as:

Step 1: Initialize weight chromosomes for control points
Step 2: Initialize knot sequence chromosome
Step 3: For each weight chromosome
3.1 For each knot chromosome
Using gradient decent learning method to find the best control points and evaluate the sum square error
3.2 Using Genetic algorithm to produce the new generation of knot chromosomes. Repeat step 3.1 and 3.2 until terminate condition is met
Step 4: Using genetic algorithm to produce the new generation of weight chromosomes. Repeat step 2 to step 4 until the terminate condition is met

4. Experiment Results

We have tried to make the experiments to approximate several kind of curves. For each curve we perform both Fixed knot RBNN and Free knot RBNN

4.1 Experiment 1 : Circle Curve Approximation

The circle curve which parametric equations x = 3*sint, y = 3*cost, is used to approximate. One hundred of points along the circle is used to train the network. We tried to approximate the curve with a rational cubic B-spline curve.

First we try to approximate the circle by RBNN with 2 segments, crossover rate 0.9, mutation rate 0.1, chromosome 14bits, population size 80. For the neural network, learning rate 0.02, momentum 0.07, the number of epochs is 200. The MSE for the best weights via epoch is given in Fig.3. The approximated curve obtained from RBNN and BNN are given in Fig.4 and Fig.5. It is easy to see that RBNN gives a better approximation than BNN. The approximation of the circle by RBNN with 3 segments is given in Fig.6

4.2 Experiment 2 : Parabola Curve Approximation

A parabola curve is also a curve from conic section. This experiment was given the parabola parametric equation as:

$$x = t^2 + t$$
$$y = 2t - 1$$

Where $t \in [-10, 10]$. The curve was replaced with 100 data points. Degree 2 of RBNN with only one segment is given in Fig.7.

4.3 Experiment 3 : Hyperbola Curve Approximation

A data curve from conic section in this experiment was a hyperbola curve with parametric equations given by

$$x = 2 * \cosh(t)$$
$$y = 3 * \sinh(t)$$

and t [-2,2]. The curve was replaced with 200 data points. The approximation by a cubic RBNN with only one segment is given in Fig. 8.

4.4 Experiment 4 : Bicorn Curve Approximation

Bicorn curve or Cocked Hat Curve can also be considered as a free form curve. Its parametric equation in this experiment is defined as :

$$x = \sin t$$
$$y = \cos^2 t\,(2 + \cos t) / 3 + \sin^2 t$$

Where t [0,2]. The curve was replaced with 100 data points. RBSNN with degree 3 and only 7 basis functions was used. The resulting curve after approximation shows in Fig.9.

4.5 Experiment 5 : Archimedean Spiral Curve Approximation

Archimedean Spiral has a shape which can be considered as a arbitray shape. Its parametric equation in this experiment is defined as :

$$x = t^{n} \cos(t)$$
$$y = t^{n} \sin(t)$$

Where t (0,2]. The curve was replaced with 200 data points. Degree 3 of RBSNN with 30 basis functions was utilized. The resulting curve after approximation shows in Fig.10.

5. Conclusion

We have developed a framework for approximate geometric object from a set of data points by RBNN. We have introduced a hybrid genetic based algorithm to training the proposed RBNN. The learning algorithm attempted automatically to optimize control points by gradient descent learning method, meanwhile optimal weights and knot (in free knot model) using genetic algorithm. The summary table 1 show that free knot RBNN model gives a smaller MSE. For regular curves as circle, hyperbola free knot RBNN improves MSE a little, meanwhile for other curves free knot RBNN gives a significantly better MSE than fixed knot model. The extension of RBNN to approximate surfaces are studied in future

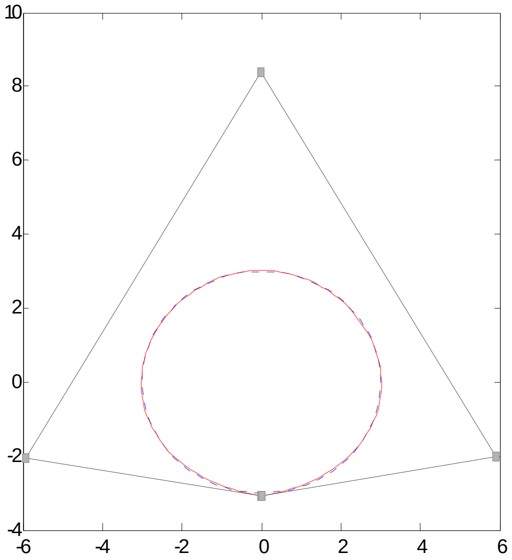

Figure 4: The resulting RBNN curve approximation and its control polygon in Experiment 1 with two curve segments

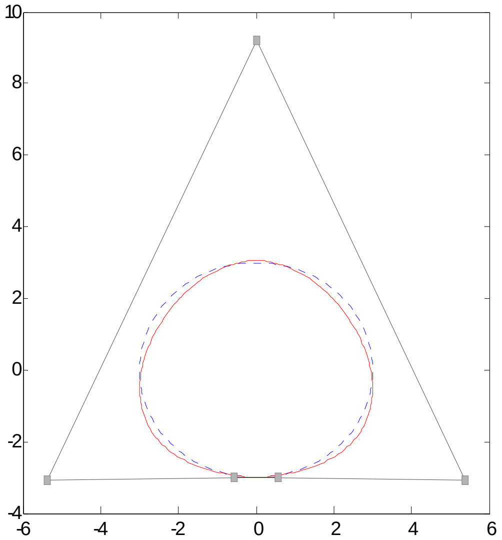

Figure 5: The resulting non-rational B-spline curve approximation and its control polygon in Experiment 1 with two curve segments

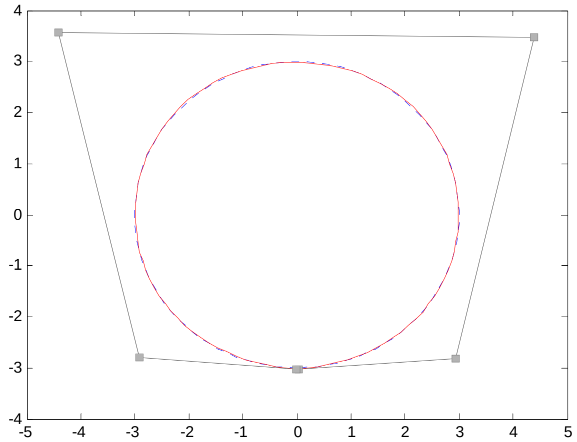

Figure 6: The resulting RBS curve approximation and its control polygon in Experiment 1 with three curve segments.

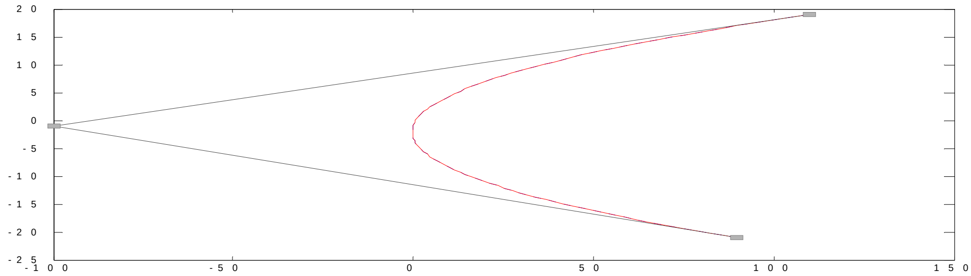

Figure 7: Approximate a parabol by a single segment of RBNN

Table 1: The comparisons between Fixed knot RBNN and Free knot RBNN

	Exp 1	**Exp 2**	**Exp 3**	**Exp 4**	**Exp 5**
Name of Curve	Circle	Parabola	Hyperbola	Bicorn	Archimedean Spiral
No. of Points	100	100	200	100	100
t (Input Range)	[-π,π]	[-10,10]	[-2,2]	[0,2π]	(0,20]
Degree of RBS	3 (cubic)	2 (quadric)	3 (cubic)	3 (cubic)	3 (cubic)
No. of Basis Functions (control points)	5	3	4	7	30
No. of Bits for Chromosome	14	14	14	14	14
Population Sizes	80	50	60	90	80
No. of generations	80	80	100	60	80
Crossover rate	0.8	0.8	0.8	0.8	0.8
Mutation rate	0.4	0.4	0.4	0.4	0.4
No. of Training Epochs	400	400	400	400	400
Learning rate	0.02	0.02	0.02	0.02	0.02
Momentum	0.7	0.7	0.7	0.7	0.7
MSE for Fixed knot RBNN	1.051e-3	4.077e-4	4.775e-5	4.319e-5	3.106e-5
MSE for Free knot RBNN	9.794e-4	4.445e-7	1.872e-6	3.135e-5	1.359e-5

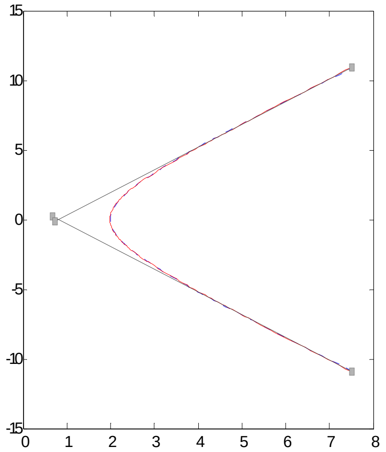

Figure 8 : Approximation a hyperbola using single segment of quadric RBNN

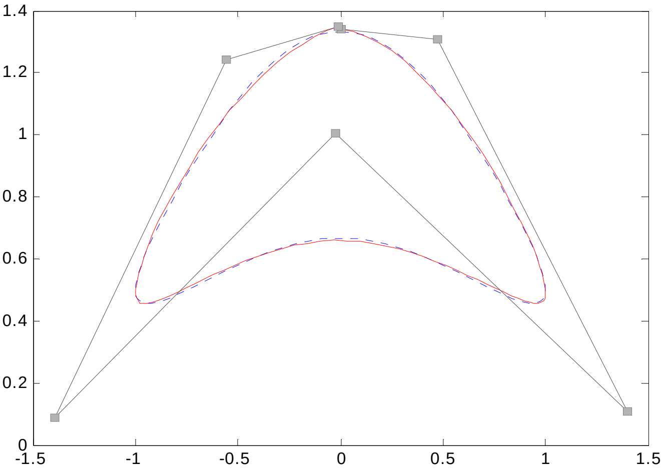

Figure 9 : Original Bicorn curve and approximation curve using RBNN

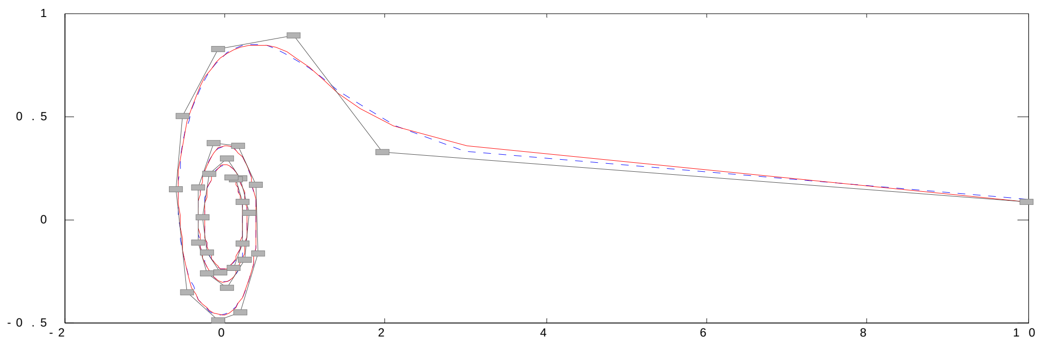

Figure 10 : Original Archimedean Spiral curve and approximation curve using RBNN

References

[1] Vien, T.D and To , T.V., 2003, **A learing framework of B-spline Neural Networks for Geometric Modeling**, INtech'03, Chiangmai, Thailand

[2] Farin, G.E, 1992, **Curves and Surfaces for Computer Aided Geometric Design: A practical Guide,** 3rd edition, Academic Press

[3] Jupp D.L.B,1978, **Approximation data by splines with free knots**. SIAM J. of Numerical Analysus, 15, 328-343.

[4] To TV and Phien HN., 1990, **Application of Polar Form to Curve Modeling**, Proc. 4th International Congress and Exhibition (Electronic Automation, 1990), New Dehli, India, pp 6-15

[5] Dierckx P., 1993, **Curve and Surface fitting with splines**. New York: Clarendon Press

[6] Schwetlick H., T.Schutze,1995, **Least square approximation by splines with free knots**, BIT 35, No.3,333-350

[7] Tet T. and To TV,2004, **Curve Matching using B-spline curve**, WSCG 04, West Bohemia, Czetch Republic

[8] Piegl, L.,1989, **Modifying the Shape of Rational B-spline Curves**, Computer Aided Design, Vol. 21, pp. 509-518

[9] Piegl, L.,1991, **On NURBS: Survey**, IEEE, CG&A, Vol. 11, pp 55-71

[10] Michalewicz Z.,19990, **Genetic Algorithm+Data Structures= Evolution Programs**, 3rd edition, , Springer Verlag, Berlin

[11] Brown, M., and Harris, C. J.,1994,,**Neurofuzzy Adaptive Modeling & Control**, Prentice-Hall, Engle-wood Cliffs, New Jersey

[12] de Boor C. and Rice, J.R, 1968. **Least squares cubic spline approximation II – variable knots**. Technical Report CSD TR21, Purdue University.

[13] Cox, M.G., 1972, **Curve and Surface fitting with splines**. New York: Clarendon Press, 1993, J. Inst. Math. Appl., Vol 10, pp 134-149

[14] Penna ,M.A.and Patterson,R.R, 1986, **Projective Geometry and Its Applications to Computer Graphics**, Prentice Hall, Englewood Cliffs