Journal of Computational Mathematics, Vol.25, No.1, 2007, 97–103.

APPROXIMATE IMPLICITIZATION BASED ON RBF NETWORKS AND MQ QUASI-INTERPOLATION *1)

Renhong Wang Jinming Wu
(*Institute of Mathematical Sciences, Dalian University of Technology, Dalian 116024, China*
Email: renhong@dlut.edu.cn, wujm97@yahoo.com.cn)

Abstract

In this paper, we propose a new approach to solve the approximate implicitization problem based on RBF networks and MQ quasi-interpolation. This approach possesses the advantages of shape preserving, better smoothness, good approximation behavior and relatively less data etc. Several numerical examples are provided to demonstrate the effectiveness and flexibility of the proposed method.

Mathematics subject classification: 41A05, 65D17, 92B20.
Key words: RBF networks, MQ quasi-interpolation, Approximate implicitization, Rational curves.

1. Introduction

As we know that parametric curves/surfaces and implicit curves/surfaces are two important topics in Computer Aided Geometry Design and Geometric Modeling. It is easy to generate points on parametric curves/surfaces. On the other hand, it is convenient to determine whether a point is on, inside or outside a given solid with the implicit treatments.

Accurate implicitization (especially surface implicitization) has not been popular in practice. This is due to the fact that the curve/surface implicitization is relatively complex and the degree of the implicit curves/surfaces is higher. Another difficulty is that implicit curves/surfaces may have unexpected components and self-intersections which lead to computational instability and topological inconsistency in geometric modeling. So finding approximate implicitization has some practical. In recent years, many authors have proposed several approaches to solve this problem. The earlier work on approximate implicitization was done by Velho et al.([6]), who presented an approximate implicitization scheme from parametric surfaces to implicit surfaces based on wavelet analysis. Sederberg et al.([4]) proposed an approach to solve approximate implicitization problem by using monoid curves and surfaces. Recently, Chen et al.([2]) presented a concept of interval implicitization of rational curves and developed an effective algorithm.

In this paper, we put forward a new method for solving the approximate implicitization problem based on RBF networks and MQ quasi-interpolation. This method has the advantages of shape preserving, better smoothness, good approximation behavior and relatively less data etc. Numerical examples are provided to illustrate the effectiveness and flexibility of the proposed method.

2. The Principle of RBF Networks

A RBF network is a three layer feed-back network consisting of one input layer, one hidden layer, and one output layer ([5]). The input layer feeds the input data to each neuron of hidden layer. Each hidden layer neuron calculates the distance between the input vector and its own

* Received December 12, 2005; accepted March 6, 2006.
1) Project supported by the National Natural Science Foundation of China (No.10271022, No.60373093 and No.60533060).

center. The determined distance is transformed via radial basis functions, and the result is exported from a neuron. Each output layer neuron is fully connected to the hidden layer and computes a linear weight sum of the outputs of the hidden neurons.

The output formula of a RBF network is calculated as follows:

$$f_j(x) = \lambda_{j0} + \sum_{i=1}^{N} \lambda_{ji}\phi(\|x - c_i\|),\ j = 1, \cdots, M, \tag{1}$$

where M denotes the number of the output layer neurons, N denotes the number of the hidden layer neurons, $x \in R^d$ denotes the input data, c_k represents the center of the kth basis function, λ_{j0} denotes the biases of the jth hidden layer neuron and λ_{ji} is the weight parameter between the ith hidden layer neuron and the jth output layer neuron.

In (1), $\phi(r)$ is a radial basis function. Examples of the radial basis functions used in applications include

(1) Gauss distribution function of Kriging method: $\phi(r) = e^{-cr^2}$;

(2) Hardy's MQ and inverse MQ functions: $\phi(r) = (r^2 + c^2)^{\frac{1}{2}}$ and $\phi(r) = (r^2 + c^2)^{-\frac{1}{2}}$;

(3) Wendland's compactly supported radial basis function.

If we choose the centers of the radial basis functions properly, any multivariate continuous function can be approximated with arbitrary precision by a RBF network with small number of neurons.

The application of a RBF network requires a training set for learning phase and a testing set for evaluating phase. For computational convenience, we adopt the following learning algorithm ([3]):

Step1. Select the centers of the radial basis functions in the hidden layer as the training points, i.e., $c_i = x_i,\ i = 1, \cdots, n = N$, where n is the number of training points. Moreover, suppose that the output layer consists of the simplest case of a single neuron.

Step2. Compute the biases and weight parameters with the RBF interpolation formula (1).

Step3. Evaluate the approximate implicitization curve with the testing point set.

3. MQ Quasi-interpolation

A radial basis function is a relatively easy multivariate function which is generated from a univariate function ([7]). Due to its simple form and good approximation behavior, the radial basis function approach has become an effective tool for multivariate scattered data interpolation during the last two decades.

Beaton and Powell ([1]) proposed a univariate quasi-interpolation formula which is the linear combination of Hardy's MQ basis

$$\phi_i(x) = \sqrt{(x - x_i)^2 + c^2}$$

and lower order polynomials. Their formula requires the derivative values at the endpoints. So it is not convenient for practical application. Wu and Schaback ([8]) gave another quasi-interpolation formula without using the derivative values at the endpoints.

Wu-Schaback's MQ quasi-interpolation formula is given by:

$$LF(x) = \sum_{j=0}^{n} F(x_j)\alpha_j(x), \tag{2}$$

where

$$\begin{aligned}
\alpha_0(x) = &\quad \frac{1}{2} + \frac{\phi_1(x) - (x - x_0)}{2(x_1 - x_0)}, \\
\alpha_1(x) = &\quad \frac{\phi_2(x) - \phi_1(x)}{2(x_2 - x_1)} - \frac{\phi_1(x) - (x - x_0)}{2(x_1 - x_0)}, \\
\alpha_i(x) = &\quad \frac{\phi_{i+1}(x) - \phi_i}{2(x_{i+1} - x_i)} - \frac{\phi_i(x) - \phi_{i-1}}{2(x_i - x_{i-1})},\ i = 2, \cdots, n-2,
\end{aligned}$$

$$\begin{aligned}\alpha_{n-1}(x) &= \frac{(x_n - x) - \phi_{n-1}(x)}{2(x_n - x_{n-1})} - \frac{\phi_{n-1}(x) - \phi_{n-2}(x)}{2(x_{n-1} - x_{n-2})},\\ \alpha_n(x) =& \quad \frac{1}{2} + \frac{\phi_{n-1}(x) - (x_n - x)}{2(x_n - x_{n-1})}.\end{aligned}$$

Theorem 3.1.([8]) MQ quasi-interpolation operator (2) preserves linear reproduction, monotonicity, convexity and variation-diminishing.

4. Approximate Implicitization of Rational Curves

Suppose that $P(t)$ is a rational parametric curve of the form

$$P(t) = (x(t), y(t)),\ t \in [a, b],$$

where $x(t)$ is a single-valued function.

Our aim is to construct $G(x, y)$ such that its zero point set $\{(x, y)|G(x, y) = 0\}$ can not only interpolate the set of training points $S = \{(x_i, y_i)|i = 1, \cdots, n\}$ of the rational curve, but also possess good global approximation behavior.

In this section, we propose a new method for solving the approximate implicitization problem based on RBF networks and MQ quasi-interpolation. The basic idea is to approximate the set of training points with MQ quasi-interpolation in order to possess shape preserving and then to approximate the error function by using RBF networks. Thus the combined curve possesses the properties of interpolation and good global approximation behavior.

Algorithm 4.1. *Approximate implicitization of rational curves.*

Input: rational parametric curve: $P(t) = (x(t), y(t)), \quad t \in [a, b]$.

Output: approximate implicit curve: $\{(x, y)|G(x, y) = 0\}$.

Step1. Select a set of training points $S = \{P_i = (x(t_i), y(t_i))|i = 1, \cdots, n\}$ and constitute an input vector matrix

$$\begin{pmatrix} x_1 & \cdots & x_n \\ y_1 & \cdots & y_n \end{pmatrix}.$$

Step2. Construct an MQ quasi-interpolation operator $LF(x)$ with data $\{P_i = (x_i, y_i)\}_{i=1}^{n}$and define an error function

$$\varepsilon(x, y) = y - LF(x).$$

Step3. Define an output vector matrix $T = (\varepsilon_1, \cdots, \varepsilon_n)$ corresponding to an input vector matrix, where $\varepsilon_i = \varepsilon(x(t_i), y(t_i))$.

Step4. Determine $f(x, y)$ by using a RBF network and the corresponding learning algorithm based on the input and output vector matrices.

Hence, $G(x, y) = f(x, y) - \varepsilon(x, y) = 0$ is the required approximate implicit curve.

Remark 4.1. The condition that $x(t)$ is a single-valued function only makes $\{x_i\}_{i=1}^{n}$ pairwise distinct, which guarantees the existence of MQ quasi-interpolation. As for general parametric curve, we can split the original curve so that $x(t)$ is a single-valued function on every segment (see Example 2). Or, we can choose the pairwise distinct training points $\{x_i\}_{i=1}^{n}$ without spitting curve whether $x(t)$ is a single-valued function or not (see Example 3 below).

It is known that any multivariate continuous function can be approximated with arbitrary precision by using sufficiently large number of hidden neurons. So we have

Theorem 4.1. *Let $G(x, y) = 0$ be the approximate implicit curve obtained from the above algorithm. Then for arbitrary $\varepsilon > 0$, we have*

$$\max_{t \in [a,b]} |G(x(t), y(t))| < \varepsilon.$$

5. Numerical Examples

In this section, we provide several examples to illustrate the effectiveness of the proposed algorithm for approximate implicitization of rational curves.

In the following figures, we simultaneously give the original parametric curve, approximate implicit curve, and the training points, denoted by black line, colored line, and black dots respectively.

Example 1. The rational curve $P(t) = (\frac{t}{1+t^2}, \frac{1-t^2}{1+t^2}),\ t \in [0,1]$.

(1) Select 10 training points and adopt Gauss function and the inverse MQ function in the hidden layer (see Fig. 1 and Fig. 2 respectively).

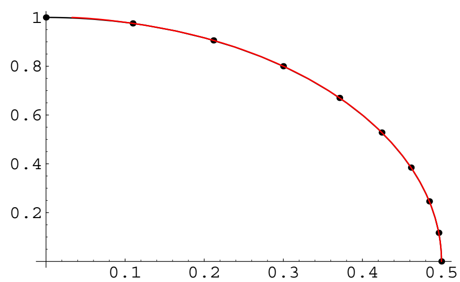

Fig. 1

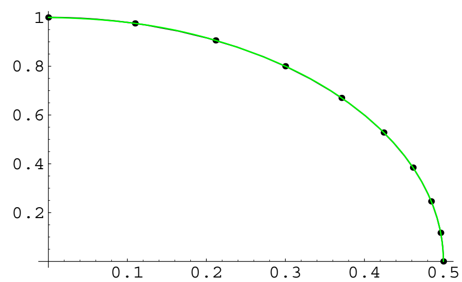

Fig. 2

(2) Select 20 training points and adopt Gauss function and the inverse MQ function in hidden layer (see Fig. 3 and Fig. 4 respectively).

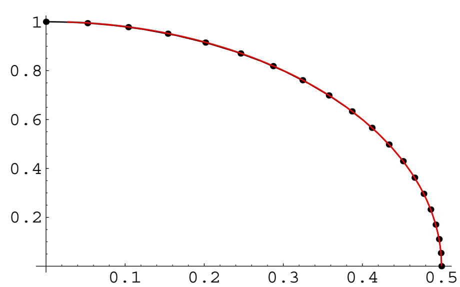

Fig. 3

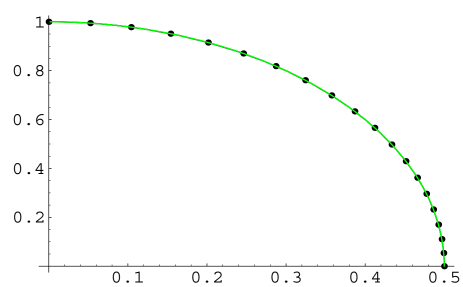

Fig. 4

Choose 100 arbitrary points as a testing set. The maximum errors and variances for the above figures are listed in Table 1.

Table 1

Figure	Fig. 1	Fig. 2	Fig. 3	Fig. 4
Max error	5.5057×10^{-4}	1.94956×10^{-3}	1.06126×10^{-4}	5.33411×10^{-5}
Variance	1.81836×10^{-7}	5.79315×10^{-7}	3.55358×10^{-6}	1.55561×10^{-10}

Example 2. The rational curve $P(t) = (\frac{2}{t^2+1}, \frac{t^3+t-1}{t^2+1}),\ t \in [-1,1]$.

It is obvious that $x(t)$ is not a single-valued function in $t \in [-1,1]$, thus we split the curve $P(t)$ at $t = 0$.

(1) Select 10 training points on every segment and adopt Gauss function and the inverse MQ function in hidden layer (see Fig. 5 and Fig. 6 respectively).

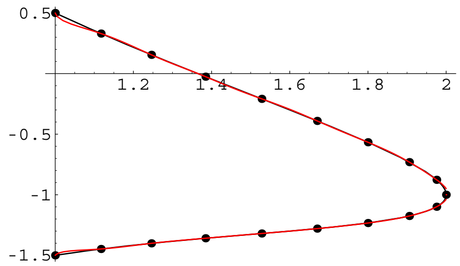

Fig. 5

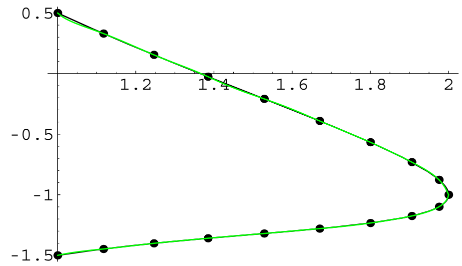

Fig. 6

(2) Select 20 training points on every segment and adopt Gauss function and the inverse MQ function in the hidden layer (see Fig. 7 and Fig. 8 respectively).

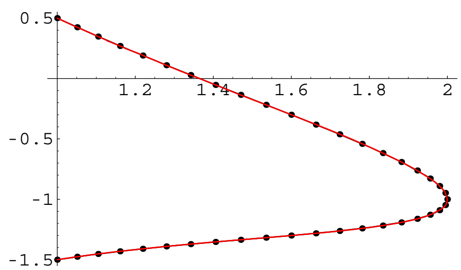

Fig. 7

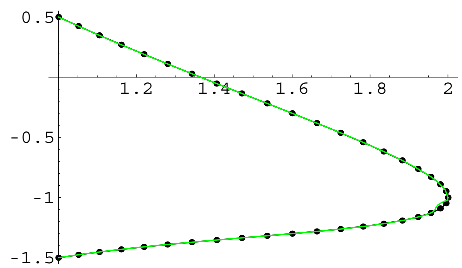

Fig. 8

Choose 100 arbitrary points as a testing set. The maximum errors and variances for Figs. 5-8 are listed in Table 2.

Table 2

Figure	Fig. 5	Fig. 6	Fig. 7	Fig. 8
Max error	1.30317×10^{-2}	7.08159×10^{-2}	5.07437×10^{-2}	2.47669×10^{-2}
Variance	1.07246×10^{-3}	3.29078×10^{-4}	4.02932×10^{-3}	2.39035×10^{-6}

Example 3. The rational curve $P(t) = (\frac{3t^3+t+1}{1+3t^2}, \frac{3t^4+t^2-1}{1+3t^2}),\ t \in [0,1]$.

Although $x(t)$ is not a single-valued function for $t \in [-1,1]$, we can still find an approximate implicit curve globally.

(1) Select 10 training points and adopt Gauss function and the inverse MQ function in hidden layer (see Fig. 9 and Fig. 10 respectively).

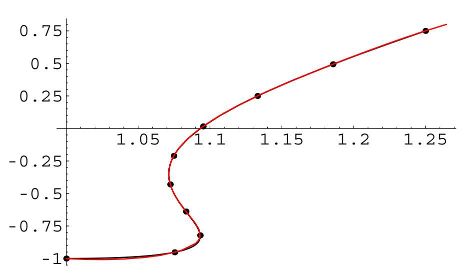

Fig. 9

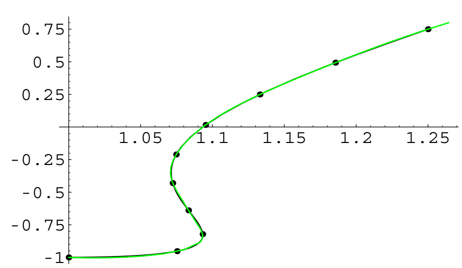

Fig. 10

(2) Select 20 training points and adopt Gauss function and the inverse MQ function in the hidden layer (see Fig. 11 and Fig. 12 respectively).

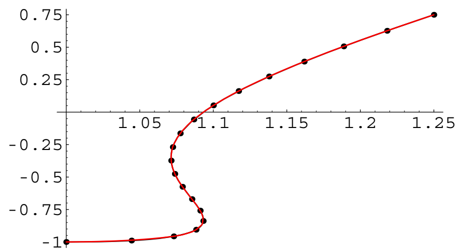

Fig. 11

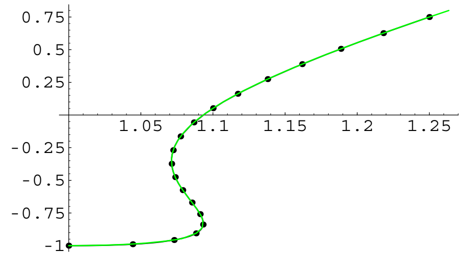

Fig. 12

Choose 100 arbitrary points as a testing set. The maximum errors and variances for Figs. 9-12 are listed in Table 3.

Table 3

Figure	Fig. 9	Fig. 10	Fig. 11	Fig. 12
Max error	966484×10^{-3}	602603×10^{-3}	3.60533×10^{-4}	1.96091×10^{-4}
Variance	5.61721×10^{-6}	3.36597×10^{-6}	2.7655×10^{-8}	3.72529×10^{-8}

Remark 5.1. From the above numerical examples, we can easily see that the parametric curve and the approximate implicit curve are almost coincide with each other, which demonstrate the effectiveness of the proposed method.

6. Conclusion

In this paper, a new method is introduced to solve the approximate implicitization problem based on RBF networks and MQ quasi-interpolation. The numerical examples demonstrate that the proposed method not only has the properties of interpolation but also possesses good global approximation behavior. Moreover, it is not sensitive to the number of training points and different types of radial basis functions. They are also satisfied with the approximate implicit curves in view of the testing effects.

It is point out that the optimal centers and radius on the radial basis functions are not investigated in this work. The problem of how to choose optimal centers and the number of radial basis functions in the hidden layer remains to be our future.

Acknowledgement. We are very grateful to the referees for their careful reading of the original manuscript and many valuable suggestions, which greatly improved the presentation of this paper.

References

[1] R.K. Beaton, M.J.D. Powell, Univariate multiquadraic approximation: quasi-interpolation to scattered data, *Constr. Approx*, **8** (1992), 275-288.

[2] F.L. Chen, L. Deng, Interval implicitization of rational curves, *CAGD*, **21** (2004), 401-415.

[3] B.T. Liao, F.L.Chen, Application of RBF networks on scattered data interpolation, *Journal of China University of Sicience and Technology*, **31** (2001), 135-142.

[4] T.W. Sederberg, J. Zheng, K. Klimaszewski, T. Dokken, Approximate implicit using monoid curves and surfaces, *Graphical Model and Image Processing*, **61** (1999), 177-198.

[5] H. Simon, Neural Network: A Comprenhensive Foundation, second edition, China Machine Press, Beijing, 2004.

[6] L. Vehlo, J. Gomes, Approximate conversion from parametric to implicit surfaces, *Computer Graphics Forum*, **15** (1996), 327-337.

[7] Z.M. Wu, The radial basis representations of functions, *Advances in Mathematics* (Chinese), **27** (1998), 202-208.

[8] Z.M. Wu, R. Schaback, Shape preserving interpolation with radial basis function, *Acta Math. Appl. Sinica*, **10** (1994), 443-448.